ANO 1 Nº 03

Cz$ 440,00

CPU

MSX 2.Ø

A PPI

O Comando Play

Comandos do MSX-DOS

100 DICAS PARA MSX

Editora Aleph

TÉCNICAS E TRUQUES DE PROGRAMAÇÃO

linguagem de máquina
ASSEMBLY Z-80
MSX
Rossini - Figueredo

Nossos livros podem ser encontrados em livrarias e lojas de computação. Se o seu livreiro ou fornecedor habitual não os tiver disponíveis, entre em contato conosco pelo telefone (011) 843-3202.

Se você não está recebendo seu boletim gratuitamente pelo correio, ou tem algum amigo que gostaria de recebê-lo, não deixe de enviar o cupom abaixo à EDITORA ALEPH - C.P. 20707 - CEP: 01498 - SÃO PAULO-SP.

NOME: ....

END.: ....

CEP: .... **CIDADE:** .... **UF:** ......

TEL: (.....) .... **MICRO(S) QUE POSSUI:** ....

ÍNDICE

SEÇÕES

CPU

Águia Informática Ltda.
Rua Santa Clara, 98/415
Copacabana
Rio de Janeiro – RJ
CEP 22041
Tel. (021) 257-4402

DIRETOR RESPONSÁVEL
Gonçalo R. F. Murteira

DIRETORIA TÉCNICA
Antônio F. S. Shalders
Carlos E. A. Moreira
André L. de Freitas
J. L. Fonseca

JORNALISTA RESPONSÁVEL
Dolar Tanus
Registro 430-RS

REVISÃO DE TEXTO

CAPA
José Aguilera

ASSINATURAS
Eduardo Simplício

ADMINISTRAÇÃO
José Newton Barros

Comando Play

EDITORA ALEPH
CURSO DE MÚSICA

FUNÇÃO

O comando PLAY DO MSX permite tocar seqüências de notas ou acordes musicais compostos de uma, duas ou três notas simultâneas, com andamento, duração, timbre ou volume programáveis.

SINTAXE:

A sintaxe do comando PLAY é

PLAY A$, B$, C$

onde A$, B$, C$ são variáveis "strings" (alfanuméricas) que contém uma seqüência de caracteres correspondentes aos sub-comandos especificados adiante. A$ é executada no primeiro canal de voz, B$ no segundo e C$ no terceiro.

A seqüência de sub-comandos pode ser definida logo após o comando PLAY, sendo digitada entre aspas, ou pode ser atribuida a uma variável string num trecho anterior do programa BASIC.

Exemplos:

1) 10 PLAY "C D E F G A B"

2) 10A$ = "C D E F G A B"
   20 PLAY A$

A variável string que determina a seqüência de sub-comandos a serem executados pelo PLAY pode ser obtida pela "concatenação" de strings menores.

Devemos ter sempre o cuidado de transformar variáveis numéricas em variáveis alfanuméricas usando a função STR$.

Digamos, por exemplo, que se queira executar a seguinte seqüência de sub-comandos:

PLAY "T 150 V 12 AB"

onde os números 150 e 12 são obtidos de variáveis numéricas definidas em outros pontos do programa. Neste caso, montaremos a variável string correspondente, conforme indica o exemplo a seguir.

```
10 X = 150
20 Y = 20
30 A$= "T" + STR$(x) + "V" + STR$(Y) + "AB"
40 PLAY A$
```

Uma outra maneira de se montar a string de sub-comandos, a partir de variáveis anteriormente definidos no programa, é a de se inserir estas variáveis com delimitadores especiais.

Quando a variável é alfanumérica, ela deve ser precedida por um "X" e seguida por ";".Quando é numérica deve ser precedida por um "=" e seguida por ";".

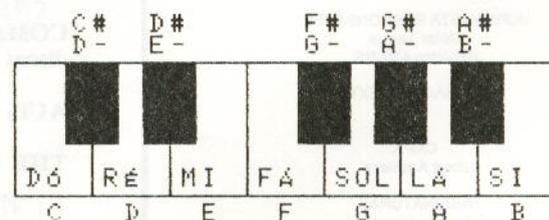
No exemplo anterior, podemos montar a string como segue:

```
10 A = 150
20 B = 12
30 Cz$ = "AB"
40 PLAY"T=A;V=B;XC$;"
```

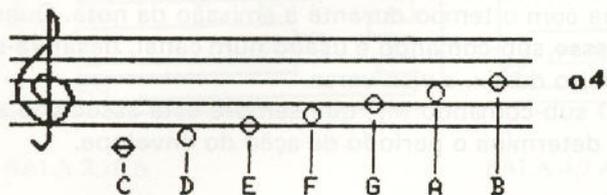
OS SUB-COMANDOS DO PLAY

Em todos os sub-comandos, que veremos a seguir, podem ser utilizadas, indiferentemente, letras maiúsculas ou minúsculas.

A – G

As letras de A à G determinam a nota a ser usada conforme as cifras da notação anglo-saxônica. Colocando um sinal # ou + logo após a letra, a altura será elevada em meio tom (sustenido). Colocando um sinal - (menos) sua altura abaixará meio tom (bemol).

Ox

Este sub-comando especifica a oitava em que se situam as notas a serem executadas. O valor de X pode estar entre 1 e 8.

Ao ser ligado o micro, o valor inicial que ele assume ("default") é de 4:

O programa a seguir executa a escala musical de DÓ a SI desde a oitava 1 (mais grave) até a 8 (mais aguda):

```
10 A$ = " C D E F G A B "
20 FOR I = 1 TO 8
30 B$ = " O " + STR$ (I)
40 PLAY B$ + A$
50 NEXT I
```

Lx

Este sub-comando determina a duração de uma nota a ser tocada. O valor de x pode estar entre 1 (duração mais longa) e 64 (mais curta).

A correspondência entre os valores de Lx e as figuras musicais é dada na figura a seguir:

A nota de maior duração é a semibreve (L1). Os valores de x indicam o denominador da fração de semibreve que a nota dura. Por exemplo: a semínima tem duração L4. Isto significa que sua duração é 1/4 da semibreve.

A duração da nota pode, também, ser especificada após a cifra. Por exemplo: L 8 A equivale a A 8. Se não houver especificação de duração após a cifra, vale a especificação no último L.

Os valores de x não precisam ser, obrigatoriamente, potências inteiras de 2 (1, 2, 4, 8, ... etc).

Isso permite, por exemplo, especificar a duração de "tercinas".
Imagine este compasso:

Como o compasso é 3/4, nele cabem 3 semínimas (3 notas de duração L4). Como o segundo tempo têm 3 notas, cada uma deverá durar 1/3 de 1/4, ou seja, 1/2. Sua duração portanto, será L 12. O compasso acima será executado por:

```
PLAY " o4 g 4 a 12 g 12 f 12 b 4 "
```
ou
```
PLAY " o4 L 12 g 4 a g f b 4 "
```
ou
```
PLAY " o4 L 4 g a 12 g 12 f 12 b "
```

Colocando um ponto (.), após a especificação da duração, esta passa a valer 1 + 1/2 vezes a duração anterior. Cada ponto adicional equivale a um acréscimo da metade do ponto anterior.

Exemplos:

A1. : duração 1 + 1/2 = 3/2 de A 1
A1..: duração de 1 + 1/2 + 1/4 = 7/4 de A 1

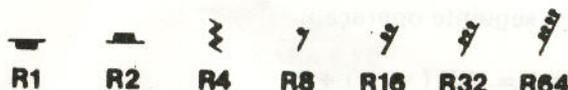
Ao ligarmos o computador o valor inicial ("default") é de L 4.

Tx

Este sub-comando especifica o andamento da música (adagio, allegro, presto, etc) corresponde à marcação de um "metrônomo interno" do MSX. O valor de x pode variar de 32 (lento) a 255 (rapidíssimo).

Ao ligarmos o computador o valor inicial ("default") é T 120, que corresponde a 120 semínimas por minuto.

Rx

Especifica a duração das pausas (silêncios). O valor de x, como no sub-comando Lx, especifica a duração da pausa, indo de 1 a 64, conforme a correspondência a seguir:

Podem ser usados valores inteiros que não precisam, necessariamente, potências inteiras de 2 (1, 2, 4, 8, 16 ...etc).

Se nada for especificado após o R, o MSX assume o valor R 4.

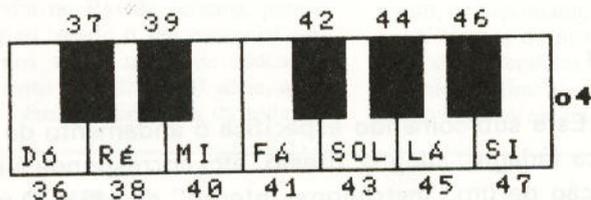
Vx

Determina o volume (intensidade) de cada canal de som do MSX. O valor de x varia de 0 (silêncio) a 15 (volume máximo). Quando usamos esse sub-comando num dos 3 canais, o MSX passa a emitir, neste canal, som de órgão, com maior ou menor intensidade, conforme o valor de Vx utilizado.

Nx

Este sub-comando representa uma alternativa à notação de cifras para especificar uma determinada nota. O valor de x pode variar de 0 a 96. Quando x=0 o sub-comando N0 corresponde a uma pausa.

De N 1 a N 96 temos todos os íons possíveis de serem obtidos pelo comando PLAY.

Um acréscimo de uma unidade ao x implica na subida de um semi-tom na altura de nota. Como existem 12 semi-tons numa oitava, somando-se ou subtraindo-se 12 ao x de Nx, obter-se-á a mesma nota uma oitava acima ou uma oitava abaixo. Na oitava 4 (valor "default" do MSX) a correspondência entre os valores de x no Nx e as notas musicais é dada na figura a seguir:

Observe que a nota mais grave do PLAY (n 1) corresponde a um do # (1 + 12 + 12 + 12 = 37) e a mais aguda (N 96) é um Si # (Dó) pois 96 - 12 - 12 -12 - 12 = 48.

Obs: Quando colocamos o MSX numa certa oitava, ao comandarmos B (SI # SUSTENIDO) ele não toca o Dó da oitava seguinte mas sim o Dó do começo da oitava em que ele se encontra. Desta forma, a única maneira de se obter o Dó da oitava 9 (que não existe no MSX) é comdarmos N 96.

Para saber a que oitava pertence a nota Nx, basta fazer a seguinte operação:

OITAVA = INT (x/12) + 1

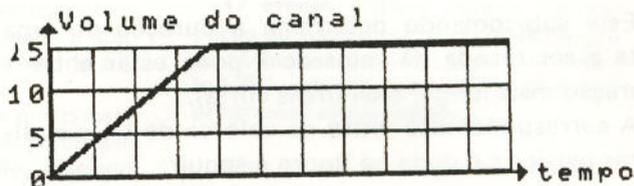
Sx e My

O sub-comando Sx permite determinar a forma do "envelope" do tom emitido, ou seja, como o volume varia com o tempo durante a emissão da nota. Quando esse sub-comando é usado num canal, desativa-se o efeito do Vx, e vice-versa.

O sub-comando My, que sempre está associado ao Sx, determina o período de ação do envelope.

Exemplificando:

O sub-comando S 13 faz com que o volume do canal, em função do tempo, varie conforme o diagrama a seguir:

Aumentando o valor de y no sub-comando My, esta variação ocorre ao longo de um período maior:

Note que, neste exemplo, apesar do volume final do envelope ser V 15, nem sempre a nota emitida atinge esta intensidade final pois sua duração pode não alcançar o final do envelope:

Por isso, a escolha do Sx e My deve ser extremamente criteriosa, caso contrário algumas notas praticamente "desaparecem" durante a execução.

O valor de y no My (período do envelope) pode variar de 0 a 65535 enquanto que o x do Sx (forma do envelope) pode variar de 0 a 15.

As formas dos envelopes que podem ser obtidas com o sub-comando Sx são dadas na figura a seguir:

VALOR ATRIBUIDO AO REGISTRO 13 (valores atribuídos a Sn da função PLAY)	FORMA DO ENVELOPE
Ø,1,2,3 ou 9	
4,5,6,7 ou 15	
8	
10	
11	
12	
13	
14	

2) A FUNÇÃO PLAY

Quando uma sequência de sub-comandos é utilizada pelo comando PLAY, o MSX não a envia diretamente ao PSG (gerador programável de sons) mas sim a uma região de memória denominada BUFFER do PSG. Feito isso, o micro-processador se considera liberado e passa a executar o resto do programa BASIC. Portanto, o PSG vai lendo e esvaziando o BUFFER, tocando a música enquanto o micro já está realizando outra tarefa. Isso pode causar efeitos indesejáveis de falta de sincronização. Para evitar isso, podemos utilizar a função:

PLAY (C)

Onde C é o número do canal que queremos testar. Se o canal estiver ativo, a função assume o valor -1, se estiver mudo, (BUFFER vazio) o valor da função será 0.

Se quisermos "congelar" o programa até que, por exemplo, o canal 2 tenha terminado sua execução, devemos usar uma linha de BASIC do tipo

```
575 IF PLAY (2) = -1 THEN GOTO 575
```

Neste caso, a execução do programa, em BASIC ficará parada na linha 575, enquanto o BUFFER do canal 2 contenha códigos a serem enviados para o PSG.

Uma vez terminada a execução da música no canal 2, o valor de PLAY (2) passa a ser 0 e o MSX passa à linha seguinte do programa.

Se o valor de C for 0, o MSX testará a execução em todos os 3 canais.

PPI Software

ANDRÉ L. F. DE FREITAS

Procurando seguir a linha de racicínio adotada nos números anteriores de CPU, onde procuro passar o máximo de informações possível a respeito do sistema MSX, continuo neste número a descrição das rotinas principais de controle do hardware MSX. Também devo este artigo à colaboração de outro autor que, em conjunto, desenvolveu uma artigo sobre o próprio hardware associado ao circuito de seleção de SLOTS e páginas de memória, com a intenção de não deixar aos usuários leitores de nossa revista sequer uma pequena dúvida sobre todo este sistema de gerenciamento de memória.

Nossos artigos, estes e os que virão futuramente, vão desvendar aos leitores todos os mistérios da poderosa máquina MSX.

A descrição PPI (Programable Peripherical Interface), circuito integrado de código 8255, será feita por um outro artigo presente neste número de CPU.

No BIOS (Basic Input/Output System) do MSX, localizado na página zero da memória do MSX, portanto contido em ROM, existem algumas rotinas utilizadas pelo próprio micro e, portanto, de simples compreensão e uso, que trabalham com todo os sistema de paginação de memória.

Estas rotinas trabalham de modo a alterar a configuração de SLOTS e páginas no MSX, chamar à execução rotinas que estejam em páginas não ativas no momento, escrever ou ler bytes em lugares não acessíveis diretamente na memória.

O próprio sistema operacional do MSX procurá, durante a função de BOOT, páginas SLOTS existentes por toda a memória, expandidos ou não.

Vamos começar a descrição destas rotinas do BIOS MSX. A descrição que se segue contém o endereço na ROM onde está localizada a rotina, sua função no sistema e respectivos parâmetros na entrada e saída destas rotinas, como valor em registradores etc. Os endereços são fornecidos em hexadecimal.

Todas as rotinas abaixo devem ser usadas com cuidado, pois valores não coerentes, como SLOTS do micro não existentes, ou endereços que não contenham rotinas, no caso de execução, podem ocasionar perda total de controle no MSX, restando ao usuário a única opção de RESET ou mesmo desligar e ligar novamente o micro, perdendo todo o trabalho.

As rotinas são utilizadas pelo próprio sistema MSX para tarefas como a leitura de um cartucho de software, ou mesmo o acesso a interfaces como Disk Drive, cartão 80 colunas, etc.

Espero satisfazer a todos aqueles que tenham curiosidade em saber onde se encontram estas rotinas e propiciar aos nossos leitores a economia de tempo tentando procurar pela memória estas rotinas, que muitas vezes é feita em desespero.

Voltaremos, em breve, com novos artigos desvendando o Hardware e Software associado para a linha MSX. Até breve.

```
Endereço: 000C - rotina RDSLT
          Lê um byte da memória em SLOT e en-
          dereço passados por registradores.
Entrada:  Reg. A - contém o SLOT ao qual será
          feita a leitura, cujos bits tem a
          forma:
            X000SSPP
          onde os bits PP identificam o
          SLOT primário (0-3)
          os bits SS identificam SLOT
          secundário (0-3)
          o bit de controle X quando possui
          valor 1, identifica que há SLOT
          secundário a acessar.
          os demais bits representados por
          000, podem conter qualquer valor.
          Ex: A - 10001101 (binário)
              significa slot primário 1
              (01), secundário existe e tem
              valor 3 ( 11 em binário ).
          Reg. HL - contém o endereço a ser
          feita a leitura (2 bytes)
          Ex: HL - 0100
              leitura na página zero (ende-
              recos 0000-3FFF)
Saída:    Reg. A - contém o valor que se
          encontrava no endereço e SLOT espe-
          cificados na entrada.
Obs:      Esta rotina altera o conteúdo dos
          registradores AF, BC e DE.
```

Endereço: 0014 - rotina WRSLT

Escreve um byte na memória, em SLOT e endereço passados por registradores.

Entrada: Reg. A - contém o SLOT ao qual será escrito o byte, cujos bits tem a forma:

X000SSPP

onde os bits PP identificam o SLOT primário (0-3)

os bits SS identificam SLOT secundário (0-3)

o bit de controle X quando possui valor 1, identifica que há SLOT secundário a acessar.

os demais bits representados por 000, podem conter qualquer valor.

Ex: A - 00000010 (binário)

significa slot primário 2 (10 em binário) sem SLOT secundário a ativar.

Reg. HL - contém o endereço a ser feita a leitura (2 bytes)

Ex: HL - 82A0

escrita na página dois (endereços 8000-BFFF)

Reg. E - contém o valor que se deseja escrever no endereço e SLOT especificados acima.

Saída: Nenhuma

Obs: Esta rotina altera o conteúdo dos registradores AF, BC e D.

Endereço: 0024H - rotina ENASLT

Habilita e mantém um SLOT da memória habilitado, de acordo com valores passados por registradores.

Entrada: Reg. A - contém o SLOT ao qual será escrito o byte, cujos bits tem a forma:

X000SSPP, de maneira semelhante às rotinas anteriores.

Reg. HL - contém o endereço que identifica a página a ser habilitada.

Saída: Nenhuma

Obs: Esta rotina altera o conteúdo de todos os registradores.

Endereço: 001C - rotina CALSLT

Executa uma rotina existente na memória, em SLOT e endereço passados por registradores.

Entrada: Reg IY - o byte mais significativo do registrador IY contém o SLOT ao qual será escrito o byte, cujos bits tem a forma:

X000SSPP

onde os bits PP identificam o SLOT primário (0-3)

os bits SS identificam SLOT secundário (0-3)

o bit de controle X quando possui valor 1, identifica que há SLOT secundário a acessar.

os demais bits representados por 000, podem conter qualquer valor.

Ex: IYH (IYH significa parte alta do reg.) - 00000000 (binario)

significa slot primário 0 (00 em binario) sem SLOT secundário a ativar.

Reg. IX - contém o endereço da rotina a ser executada

Ex: IX - C100

rotina na página três (endereços C000-FFFF)

Saída: Dependente da rotina executada.

Obs: Os valores nos registradores podem ser alterados de acordo com a rotina que for executada.

Endereço: 0030 - rotina CALLF

Executa rotinas contidas em outros SLOTS.

Modo de acesso: RST 30H

DB Identificação do SLOT

DW Endereço da rotina a executar

Obs: O byte identificador de SLOT tem a forma X000SSPP, como já descrito anteriormente.

Os dois bytes de endereço identificam a localização na memória da rotina executada.

O retorno desta chamada e registradores modificados dependem da rotina executada.

Funções do Basic

ANTÔNIO F. S. SHALDERS

O objetivo desta coluna é de apresentar, todos os meses, uma série de comandos e/ou intruções do MSX/BASIC.

Como esta coluna é voltada para o iniciante, todo o possível será feito no sentido de ensinar ao mesmo como utilizar corretamente o MSX-BASIC. Quando necessário, as explicações serão acompanhadas de um programa exemplo com o fim de esclarecer alguma eventual dúvida.

A FUNÇÃO ABS:
Sintaxe: ABS (EXPRESSÃO)

Fornece o valor absoluto do argumento, que pode tanto ser uma expressão matemática como um símbolo de valor numérico.

Exemplo: ABS (10) = 10
ABS (-15) = 15

A listagem 1 contém um programa que demonstrará o traçado das funções seno e módulo do seno, de zero a PI, sendo que o valor de PI é obtido a partir da função ATN, que será comentada no decorrer do artigo.

A FUNÇÃO ASC:
Sintaxe: ASC (STRING)

A função ASC fornece o código ASCII do caracter em questão. No caso do argumento ser uma cadeia de caracteres, somente o primeiro caracter será considerado.

Exemplo: ASC:("A") = 65
ASC("A") = 65

Se A$="CPU", ASC(A$) = 67, pois 67 é o código do caracter 'C'.

A FUNÇÃO ATN:
Sintaxe: ATN (EXPRESSÃO)

Fornece o arco-tangente da expressão ou ângulo em questão. O argumento deverá estar, obrigatoriamente, em radianos.

Exemplos: T = TAN(PI/4)
T = TAN(.7853981634)
T = 1
ATN(T) = .7853981634

Uma forma bastante prática de obtermos o valor de PI é usando a função ATN:

PI = 4 * ATN(1)

Outros múltiplos de PI podem ser achados da mesma forma. Convém lembrar que PI radianos equivalem a 180 graus.

A INSTRUÇÃO SCREEN

Ao contrário da mencionada acima, a instruÇão SCREEN é uma das mais poderosas do micro.

Esta instrução controla o modo de operação do vídeo, as dimensões dos sprites, o estalido das teclas, a velocidade de gravação em fita, e, finalmente, o tipo de impressora acoplado ao seu micro.

SINTAXE: SCREEN A, B, C, D, E

A: Modo de operação do vídeo (0, 1, 2 ou 3)
B: Dimensão dos sprites
0 – sprites 8 x 8
1 – sprites 16 x 16
2 – sprites 16 x 16
3 – sprites 32 x 32
C: Estalido das teclas
0 – desativado
1 – ativado
D:Velocidade de gravação
0 – 1200 BPS
1 – 2400 bps
E: Tipo de impressora
0 – MSX
1 – ABICOMP

O COMANDO AUTO:
Sintaxe: AUTO X, Y

Faz a numeração automática das linhas de programação. É o conteúdo "default" da tecla F2 do seu MSX.

Os parâmetros X e Y são opcionais e representam o número de linha inicial e o incremento, respectivamente.

Exemplos:
AUTO
Inicia a autonumeração na linha 10, com incremento de 10.

AUTO 50
Idem, porém com início na linha 50.

AUTO ,1
Inicia a autonumeração na linha 10, com incremento de 1.

AUTO 100,5
Inicia a autonumeração na linha 100, com incremento de 5.

AS FUNÇÕES SPACE$(X) E SPC(X)

As duas funções não são iguais, como pode parecer à primeira vista, pois a SPC(X) é específica para uso na tela, enquanto que a SPACE$(X) é normalmente usada para atribuir à uma váriavel string uma cadeia de caracteres. Devem ser usadas sempre que possível, pois, além de tornarem o programa muito mais elegante, economizam muita memória. O argumento destas funções indica o número de espaços em branco desejado.

A VARIÁVEL DO COMANDO BASE:

É uma coisa não muito utilizada, quer por não ser conhecida, quer por sua pouca utilidade prática, pois esta variável fornece determinados valores, dependendo do argumento em questão, podendo ser substituída por uma tabela. Aliás esta tabela vem no próprio manual.

Esta variável está relacionada com o VDP (processador de vídeo) e sua utilização é simples.

SINTAXE: BASE(X)

X:	SIGNIFICADO:
0	Base da tabela de nome SCREEN 0
1	Sem significado
2	Base da tabela de padrões da SCREEN 0
3,4	Sem significado
5	Base da tabela de nomes da SCREEN 1
6	Base da tabela de cores da SCREEN 1
7	Base da tabela de padrões da SCREEN 1
8	Base da tabela de atributos dos sprites, SCREEN 1
9	Base da tabela de padrões dos sprites da SCREEN 1
10	Base da tabela de nomes da SCREEN 2
11	Base da tabela de cores SCREEN 2
12	Base da tabela de padrões da SCREEN 2
13	Base da tabela de atributos dos sprites, SCREEN 2
14	Base da tabela de padrões dos sprites da SCREEN 2
15	Base da tabela de nomes da SCREEN 3
16	Sem significado
17	Base da tabela de padrões da SCREEN 3
18	Base da tabela de atributos dos sprites, SCREEN 3
19	Base da tabela de padrões dos sprites da SCREEN 3

A 1ª aventura de Modesty Basic

JOÃO L. F. DE FREITAS

Aquele parecia ser mais um dia tranqüilo, como todos os outros 389 dias anteriores, desde que ela abriu a agência de DELETEtives. Modesty BASIC estava digitando alguma coisa em seu honesto MSX, quando seu assistente Dick TRACE entrou no escritório com duas marmitas. Já passava de meio dia e ela nem reparara.

Enquanto almoçavam, foram interrompidos pelo SOUND do telefone. Do outro lado da linha, uma voz se fazia ouvir em três canais de som:

– Modesty BASIC? Aqui é Robert ERASE! Eu vi seu telefone em um DATABASE que adquiri recentemente, e estou precisando de seus serviços . Venha à minha casa. Não posso falar mais por telefone. BEEP!

É claro que ela já ouvira falar do Sr. Robert ERASE. Pegou rapidamente, sua pistola MOUSE e partiu à mansão do Sr. ERASE. Dick também foi.

Quando chegaram, foram recebidos por um mordomo mal-encarado, que segurava um cachorro pela coleira, onde uma plaquinha dizia: HEX$!

– Queiram acompanhar-me, por favor. – disse o sujeito – O Sr. ERASE os aguarda.

Foram, então, levados à bilbioteca, onde um senhor de meia idade lia um manual sobre um DATABASE. Além disso, as estantes continham livros velhos e raros escritos em FORTRAN.

– Modesty! Enfim você chegou. MAXFILES, deixe-me a sós com a detetive e seu assistente!

Após o mordomo sair, BASIC perguntou ao Sr. ERASE o que havia acontecido, e ele começou a explicação:

– Semanas atrás eu terminei um sistema que venho desenvolvendo há anos, cuja função era proteger os programas contra qualquer tipo de pirataria. Porém hoje de manhã, eu descobri que o programa havia desaparecido.

– Mais alguém sabia deste programa? – perguntou Modesty BASIC.

– Apenas minha esposa Mary LINE.

– E onde está ela? – perguntou Modesty BASIC.

Neste momento, uma senhora de cabelos ruivos, vestindo um casaco de peles de COLOR 14, entrou na biblioteca.

– Sra. Mary LINE eu suponho. – falou Modesty, sem obter resposta.

– Minha esposa é muda! – disse ERASE.

– O criminoso deve ter deixado alguma HOT pista! – falou Dick TRACE, excitado.

– Não creio. Porém ele esqueceu de levar o manual de instruções! – comentou o Sr. ERASE.

– Venha, Dick, vamos interrogar os empregados! – chamou Modesty.

E assim o fizeram. Todos na mansão tiinham ar suspeito, até mesmo a cozinheira e o jardineiro, um MERGEnal redimido. Modesty havia notado que não chegaria a lugar algum sem um bom plano.

Resolveu reunir todos, inclusive Sr. e Sra. ERASE na sala de jantar. Colocou todos a par do desaparecimento do programa e da existência do manual, tentando obter uma expressão suspeita do culpado. Não adiantou. Sr. ERASE estranhou sua atitude, mas Modesty tinha um plano!

Estaria ela querendo que o criminoso roubasse o manual?

Sim! Fazia parte do plano. O culpado cometeria o ERRO de SINTAXE ao voltar ao local do crime para completar o trabalho.

À noite, ele e TRACE voltaram a biblioteca. Modesty escondeu-se atrás das estantes e TRACE ficou atrás das cortinas.

Ficaram ali por algumas horas, até que um vulto de capuz, oculto pela escuridão, adentrou o recinto.

Modesty apertou o botão da luz e descobriram, então, o culpado.

– Sra. Mary LINE!!!– exclamou Modesty, já com sua famosa pistola na mão.

Sr. ERASE, com o barulho ouvido, de pijamas, entrou na biblioteca, com uma expressão de surpresa.

– Mas como... – e antes de desperdiçar mais palavras, Modesty o interrompeu.

– Bem que eu já havia visto este rosto antes – falou Modesty, tirando a peruca da Sra. LINE, ou melhor – Vera KILL!

Programando a impressora

ANTÔNIO F. S. SHALDERS

Note que é possível programar a impressora para imprimir caracteres combinados, como o tipo "expandido-comprimido em índice", por exemplo. Basta pressionar a seqüência correta de teclas.

O programa foi desenvolvido e testado num Hotbit e em uma impressora ELEBRA, EI-8011 Mônica de 80 colunas. Para adaptá-lo a outras impressoras, basta alterar as linhas de vão de 1110 a 1240 conforme a marca da impressora em questão.

O utilitário Types destina-se aos usuários de impressoras compatíveis com a EI-6011 (Mônica). Este utilitário, de construção simples e de fácil uso, tem como finalidade fazer, confortavelmente, a seleção dos tipos de caracteres da impressora e suas possíveis combinações.

O programa é apresentado na forma de menu e, para selecionar o tipo (ou combinação de tipos), basta pressionar a(s) tecla(s) em questão. Note que a numeração das teclas foi feita em hexa, pois o objetivo principal de CPU é explorar ao máximo todos os recursos do seu MSX. Foram usadas outras funções que não são corretamente exploradas pela grande maioria dos usuários da linha MSX: as instruções INPUT$ E OUT.

No caso deste programa, a instrução INPUT$ tem uma grande vantagem sobre a clássica INKEY$, pois não é necessário uma varredura constante do teclado.

Outra coisa interessante é a comparação lógica de strings, feita na linha 1080. É claro que seria mais simples verificarmos o valor ASCII da variável A$, porém, com o procedimento adotado, economizamos o uso de um ASC, o que é prefeitamente justificável.

O OUT 170,26 liga o LED indicador de maiúsculas e o OUT 170,90 o desliga. A trava de maiúsculas é obtida através do POKE &HFCAB,1 e é desativada com POKE &HFCAB,0.

```
1000 ' TYPES 1.00
1010 ' ANTONIO FERNANDO SHALDERS
1020 ' REVISTA CPU
1030 ' ------------------------
1040 '
1050 COLOR15,1:SCREEN 0:KEY OFF:WIDTH 40
:POKE &HFCAB,1:OUT 170,26
1060 PRINT "TYPES  Ver 1.00     /    ELEB
RA  EI 6011":PRINT:PRINT
1070 PRINT"[0] STANDARD",,"[1] COMPRIMID
O 1","[2] COMPRIMIDO 2","[3] QC",,"[4] E
XPANDIDO",,"[5] INDICE",,"[6] EXPOENTE",
,"[7] SUBLINHADO","[8] AJUSTE DE ENTRELI
NHA","[9] AJUSTE DE COLUNAS","[A] TESTE"
,,"[B] FIM",,,,">> OPÇÃO: ";
1080 A$=INPUT$(1):IF (A$<"0" OR A$>"9")
AND (A$<"A" OR A$>"B") THEN 1080
1090 A=VAL("&H"+A$)+1
1100 ON A GOTO 1110,1120,1130,1140,1150,
1160,1170,1180,1190,1220,1250,1260,1080
1110 LPRINT CHR$(27);"@":GOTO 1050
1120 PRINT"CP1 ";:LPRINT CHR$(15);:GOTO
1280
1130 PRINT"CP2 ";:LPRINT CHR$(30);"5";:G
OTO 1280
1140 PRINT"QC ";:LPRINT CHR$(27);"G";:GO
TO 1280
1150 PRINT"EXPD ";:LPRINT CHR$(14);:GOTO
 1280
1160 PRINT"IND ";:LPRINT CHR$(27);"S";CH
R$(1);:GOTO 1280
1170 PRINT"EXPT ";:LPRINT CHR$(27);"S";C
HR$(0);:GOTO 1280
1180 PRINT"SUB ";:LPRINT CHR$(27);"-";CH
R$(1);:GOTO 1280
1190 PRINT:INPUT"ENTRELINHA:";E
1200 IF E<1 OR E>255 THEN E=13
1210 LPRINT CHR$(27);"A";CHR$(E);:PRINT"
EL=";E;" ";:GOTO 1280
1220 PRINT:INPUT"COLUNAS:";C
1230 IF C<1 OR C>80 THEN C=80
1240 LPRINT CHR$(27);"Q";CHR$(C):PRINT"N
C=";C;" ";:GOTO 1280
1250 PRINT"TESTE ";:FOR X=32 TO 126:LPRI
NT CHR$(X);:NEXT:LPRINT:LPRINT:GOTO 1280

1260 PRINT:PRINT"TEM CERTEZA (S/N)":A$=I
NPUT$(1):IF A$<>"S" THEN 1050
1270 CLS:OUT 170,90:POKE&HFCAB,0:NEW
1280 A=13:GOTO 1100
```

MSX 2.Ø por transformação

PEDRO HENRIQUE GAMA

Num período em que muita coisa no Brasil se tornou impraticável devido a seus custos e ao poder aquisitivo da população cada vez mais corroído pela inflação, descobre-se que é possível, através de uma adaptação no Hardware do MSX, possuir um dos mais badalados e versáteis micros do Japão e da Europa: o MSX 2.

Antes de mais nada, devo esclarecer que não é de intenção minha, neste artigo, dizer-lhes se é ou não viável financeiramente transformar o seu MSX num MSX 2. Só você e sua conta bancária podem decidir. O que farei é contar-lhe sobre minha experiência com a máquina e seus recursos. Qualquer dúvida pode ser enviada à redação de CPU, que responderei com prazer.

O que anda gerando muita dúvida sobre esta tão famosa transformação é que ninguém, ou quase ninguém, sabe o que realmente muda em seu micro durante a adaptação. Muito pouca gente sabe quem realmente o faz e, geralmente, a primeira informação que se tem a respeito é o preço da brincadeira, o que quase sempre desanima o futuro dono de um MSX 2. Este ex-futuro dono, porém, nunca viu o funcionamento de uma destas máquinas. Elas não existem, ainda, em nossas lojas para demonstrações nem em anúncios de revistas ou TV. Muitos ainda crêem piamente que a chegada deste será a ruína do já consagrado MSX standard, provocando, portanto, outra reviravolta no cenário já tão conturbado da micro informática brasileira.

A realidade é que o MSX 2 é totalmente compatível com o MSX 1 e em termos de Software e Hardware. O drive que você usa no seu Expert ou Hotbit tem de funcionar perfeitamente num MSX 2. O mesmo acontece com o data-corder, o modem etc. A diferença está no que ele faz a mais. Veja as especificações mais adiante.

Mas e daí? Qual eu devo adquirir: o importado ou o transformado? Se você pensar no fator custo, nenhum dos dois. O MSX 2 importado costuma sair para o usuário final em torno de US$500, isto em configuração mínima, sem teclado destacável ou 256 Kbytes de RAM. O mais barato se assemelha em muito com um Hotbit. É o PANASONIC da linha AV ou HITIBIT da SONY. O MSX 2 transformado sai em torno de 60 OTNs em configuração mínima, a expansão de 256K bytes sai por volta de 37 OTNs.

E os riscos? Os riscos são claros. Até o momento é terminantemente proibida a entrada de componentes ou micro computadores importados no país. Você também pode perder o seu dinheiro como pode ser processado, sem falar do perigo de se ficar na mão de um muambeiro caso você não possa viajar para comprar um da assistência técnica que não existe. Este, pelo menos, é um ponto de enorme vantagem do micro transformado, já que os seu micro tem, ou deveria ter, nota fiscal e os componentes usados são adquiridos em qualquer boa loja de eletrônica. A provável desvantagem fica por conta do fato de que no seu MSX 1 você já gastou uma grande soma em dinheiro, pelo menos para comprá-lo. Mas será que esta é realmente uma desvantagem? Vamos analisar, adiante, alguns pontos.

O micro importado não acentua. Já o nacional, por ser derivado de um MSX Br., acentua normalmente e continua com a mesma compatibilidade com sua impressora e editores de texto. Além disto, 99% dos micros importados não têm saída para TV, só para monitor monocromático ou RGB. Você sabe quanto custa um RGB? É viável usar um micro de 512 cores num monitor monocromático? O micro importado não é compatível com o padrão de cores das TVs e monitores RGB brasileiros. Caso você transforme sua TV em RGB terá de transformar também seu padrão de cores, ou o do micro!

Partindo deste princípio, observamos que o MSX 2 importado serve, e muito mal, somente para o lazer e uso de jogos. Eu pelo menos, considero inadmissível fazer uma carta para os amigos sem os devidos acentos e cê-cedilhas, nem que sejam no lugar errado. Quanto aos jogos, eu os adoro, mas com precisão de imagens e cores.

Até o momento, já podemos observar, pelo menos, uma coisa: o MSX 2 importado é inviável. Pelo menos para aqueles que costumam usar bem aquilo que compram e que não pretendem gastar o preço de um IBM em todas as "transformações" necessárias para que o micro importado funcione perfeitamente. Mas, observe bem, esta é uma opinião minha. É uma opinião de quem gastou meses a procura de um importado e, quando conseguiu, se decepcionou barbaramente e não sossegou enquanto não se livrou dele. É, ainda, a opinião de um usuário que não vende seu transformado por dinheiro algum, (e olha que propostas não faltaram). É verdade, porém, que existem algumas excessões: um PHILIPS, por exemplo, que já vem com 256K bytes, digitalizador de imagens, dois drives, monitor RGB, etc. Um micro destes custa por volta de US$ 1.800, dinheiro suficiente para ir à copa da Itália em 1990 sem muitos gastos, é claro.

A grande verdade, agora, é que você pode pode ter um micro deste gabarito sem se colocar contra a lei. Muitas pessoas pensam que esta transformação é uma coisa "caseira", o que não é verdade. Vários dos circuitos usados na transformação são feitos em dupla face e encomendados em indústrias específicas.

E então, é viável? Analise bem as suas necessidades quanto a um micro computador. Se você trabalha com gráficos e cores, certamente irá considerá-lo um micro incrível.

Se a sua necessidade é somente trabalhar com sons, esqueça toda aquela balela de que MSX 2 tem mais canais de sons do que o MSX 1. Quanto ao resto, facilidade de uso, mais comandos e recursos, você encontrará aos montes no seu MSX 2. Tire, então, as suas conclusões, verifique as suas disponibilidades financeiras e siga um único conselho: antes de abrir a carteira para adquirir o seu MSX 2, seja ele qual for, pegue todas as informações possíveis sobre procedência, especificações técnicas e garantia.

MSX 2 - CARACTERÍSTICAS REAIS (SEM ESPECULAÇÕES)

CONFIGURAÇÃO MÍNIMA

ORIGINAL IMPORTADO

MICRO PROCESSADOR - Z-80A
RAM - 64K Bytes RAM
VRAM - 64K Bytes RAM
ROM - 48K Bytes
CAPACIDADE DE TEXTO -
80 caracteres x 24 linhas
RESOLUÇÃO GRÁFICA -
512 x 212 Pontos
CORES - 16/512 ou 256
INTERFACE DE SOM -
8 oitavas
3 Canais de Áudio
1 Canal de Ruído
ACENTUAÇÃO? - Não
Saída para TV - Não
Saída para RGB - Sim
Saída Monitor? - Sim
Bateria interna? - Sim
Padrão de Cores - NTSC
PAL-G
Outros dados ídem MSX 1

TRANSFORMAÇÃO BRASILEIRA

MICRO PROCESSADOR - Z-80A
RAM - 64K Bytes RAM
VRAM - 128K Bytes RAM
ROM - 48K Bytes
CAPACIDADE DE TEXTO -
80 caracteres x 24 linhas
RESOLUÇÃO GRÁFICA -
512 x 212 Pontos
CORES - 16/512 ou 256
INTERFACE DE SOM -
8 oitavas
3 Canais de Áudio
1 Canal de Ruído

ACENTUAÇÃO? - Sim (MSX Br.)
Saída para TV - Sim
Saída para RGB - Sim
Saída Monitor? - Sim
Bateria interna? - Sim
Padrão de Cores - PAL-M
Outros dados ídem MSX 1

Não acompanha vários importados: cabo para Data-Corder ou RGB

O que se pode solicitar na hora de fazer a transformação (somente se feita diretamente com o Ademir)

- Inclusão de uma expansão de 256K bytes
- Inclusão de um botão de RESET real no painel do EXPERT
- Inclusão de interface interna para 1 a 4 Drives
- Inclusão de 1 ou 2 Drives 3,1/2, 5,1/4 (ou ambos) no Gabinete
- Inclusão de um programa Assembler/Disassembler residente
- Inclusão de um digitalizador de imagens (Breve)
- Transformação de sua TV a CORES em RGB para MSX,
- Outras personalizações podem ser feitas em caráter especial e sob encomenda.

* = características do modelo testado.

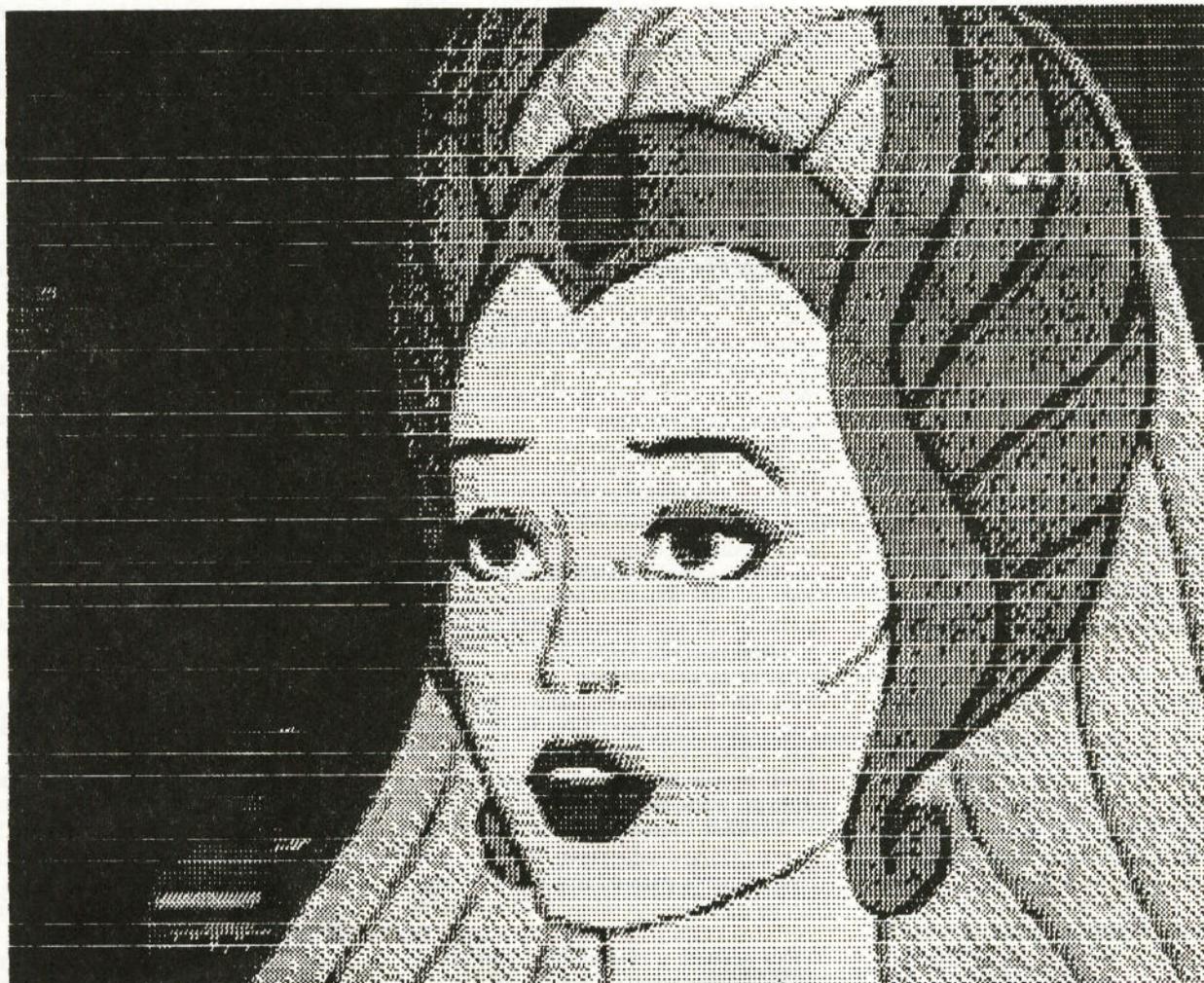



IMAGEM DIGITALIZADA PELO MSX 2.0

Técnicas de overlay

ANTÔNIO F. S. SHALDERS

Overlay é o termo que significa que um programa deverá ser dividido em duas ou mais partes a fim de que este caiba na memória do micro. Cada uma dessas partes é utilizada apenas quando necessário, sendo armazenada normalmente em disco. Também podem ser feitos overlays utilizando os slots. A grande maioria dos jogos em linguagem de máquina usa este método, pois é comum vermos um programa principal dividido em 6 ou mais subprogramas de 16kb de extensão.

Esta técnica era de uso praticamente obrigatório até o final da década de 70, pois os computadores desta época não tinham tanta memória disponível.

Um caso típico de overlay para o MSX é no Turbo Pascal 3.0, da Borland. Neste pacote, há um pequeno arquivo chamdo TURBO.OVR. A extenção para um programa de overlay, é justamente OVR. Outros casos são .XXX, onde XXX é o número do overlay. É comum vermos, por exemplo, TURBO.COM seguido de TURBO.OVR ou então TESTE.001, TESTE 002, e assim por diante.

Existem casos em que não podemos abrir mão desta técnica em BASIC, pois um programa em BASIC não pode ter mais do que aproximadamente 28 kb de texto e dados. Se este limite tiver que ser ultrapassado, não há outra solução.

O Aztec C, que é um compilador C, utiliza esta técnica durante a compilação, a montagem do programa .ASM e a montagem do programa .COM.

Neste compilador, como os programas que executam essas tarefas não são exatamente pequenos e, em geral, os programas que estão sendo compilados também não, o jeito foi dividir o Aztec em três subprogramas: o compilador propriamente dito, o assemblador e o linker. Isto sem contar o editor de textos e o fato de que existem dois programas compiladores no mesmo pacote: um para 8080 e outro para Z-80.

Uma coisa interessante é que o MSX-BASIC tem uma intrução que pode fazer pequenos overlays sem que o programa principal seja destruído. Essa instrução é o MERGE (ou fusão de arquivos). O único cuidado a ser tomado é que os números das linhas do programa a ser fundido com o programa mestre devem ser diferentes das do programa principal. Não que isso seja obrigatório, mas pode poupar um bocado de dor de cabeça. No caso de MERGE ser utilizado para este fim, deve ser usado em conjunto com DELETE.

Um outro método é simplesmente desviar o programa principal para uma linha que tenha algo como um RUN "PROG.OVR". O programa é então substituído por outro e, para retornar ao programa principal, algo semelhante deverá ser feito. Este método é usado somente quando os programas não podem ser partidos e são muito grandes. Um cuidado especial deverá ser tomado com as variáveis, pois um RUN as apaga. Caso existam, estas deverão ser salvas temporariamente em disco ou pokeadas na própria memória do micro em endereços convenientes.

Uma das aplicações mais clássicas de overlays está nos arquivos de auxílio de um determinado programa, os .HLP, por exemplo, usados no SUPERCALC e no SCED. Para quem não sabe, o SUPERCALC é uma planilha eletrônica e o SCED é um poderoso processador de textos. Quando um help é pedido, o programa se desvia para o overlay relativo ao programa help.

Todos os casos de overlays são muito específicos, de modo que a elaboração de programas exemplo para todos os casos é tarefa quase impossível. Os programas que acompanham este artigo mostram como fazer o overlay por eliminação e são bastante primitiivos. O overlay por substituição de blocos do programa principal (usando MERGE) já é um pouco mais complexo, mas não é difícil, sendo apenas muito particular à cada programa.

O programa 1 deverá chamar-se OVERLAY.BAS e o 2, PROG.OVR.

```
100 '*********************************
110 '*                               *
120 '*  TECNICAS DE OVERLAY EM BASIC *
130 '*                               *
140 '*   Antonio Fernando Shalders   *
150 '*                               *
160 '*     Revista CPU - 1988        *
170 '*                               *
180 '*********************************
190 '
200 ' PROGRAMA BASE
210 '
220 CLS:KEY OFF
230 PRINT"PROGRAMA BASE"
240 LOCATE 0,5:PRINT"PRESSIONE UMA TEC
LA PARA O OVERLAY"
250 A$=INPUT$(1)
260 RUN "PROG.OVR"
```

```
100 '*********************************
110 '*                               *
120 '*  TECNICAS DE OVERLAY EM BASIC *
130 '*                               *
140 '*   Antonio Fernando Shalders   *
150 '*                               *
160 '*     Revista CPU - 1988        *
170 '*                               *
180 '*********************************
190 '
200 ' PROGRAMA OVERLAY
210 '
220 CLS
230 PRINT"ESTE E O PROGRAMA OVERLAY"
240 PRINT:FOR X=32 TO 255:PRINT CHR$(X)
;:NEXT
250 PRINT:PRINT:PRINT"ESTE PROGRAMA EST
A NO LUGAR DO PROGRAMA BASE ORIGINAL."
260 PRINT:PRINT"PRESSIONE QUALQUER TEC
LA PARA CARREGAR  NOVAMENTE O PROGRAMA
BASE"
270 A$=INPUT$(1)
280 RUN"OVERLAY.BAS"
```

PPI Hardware

CARLOS E. A. MOREIRA

Vamos, neste artigo, mostrar um pouco mais sobre a estrutura interna do MSX. É quase certo que um dia já tivemos algum tipo de contato com o termo PPI. Mas, afinal, o que vem a ser a tal PPI do microcomputador?

PPI é uma abreviação de Programable Peripheral Interface (Interface programável de periféricos). Isto siginifica que dentro do nosso micro MSX temos uma interface programável, fisicamente representada pelo "chip" 8255 da INTEL.

Quando o computador faz a inicialização do sistema, rotina a qual denomina-se "power-up", ele faz a programação da PPI da forma conveniente a qual ele deseje usar. Esta programação é feita basicamente dando outś em quatro portas, as quais são reservadas ao uso da PPI. Vamos descrever cada uma delas e suas funções desempenhadas. Nos reservaremos, aqui, a descrever somente a parte de hard, tendo em vista que neste mesmo número outro artigo descreve o soft da PPI.

O primeiro registrador da PPI é o "A". Este registrador é acessado através da porta A8H. Isto significa que, se quisermos ler ou escrever aí, devemos fazêlo usando "IN" ou "OUT", conforme a necessidade, na porta A8H. O registrador "A" é responsável pela configuração em que se encontra distribuído dentro dos "slots" os de 64 Kbytes de memória (rom e ram) que podem ser acessados diretamente pelo Z80. Os bit's 0 e 1 deste registrador indicam em qual slot se encontra a página zero da memória; os bit's 2 e 3 indicam em que slot está disponível a página 1; as páginas 2 e 3 são identificadas pelos bit's 4 e 5, 6 e 7 respectivamente.

O segundo registrador é o "B", que é acessado através da porta AH9, tem a função de assessorar o teclado do micro.

Como é sabido, o teclado do MSX é uma matriz, 10x8. Desta forma, o registrador "B" fica incumbido de fazer a leitura das oito colunas desta matriz, cada uma delas associadas a um bit deste registrador.

O terceiro registrador é o "C", que tem acesso através da porta AAH. Sobre este falarei com mais carinho, especialmente sobre os quatro bit's mais significativos, já que aí se encontra o objetivo maior deste artigo. Os quatro bit's menos significativos deste registrador, juntamente com a porta "B", fazem assessoramento do teclado. A FIGURA 1 nos dá uma idéia melhor de como é feita a decodificação do teclado usando todos os oito bit's do registrador "B" da PPI e os quatro bit's menos significativos do registrador "C".

Antes de dar prosseguimento, devo falar da porta "D" da PPI. Esta porta indica o modo de operação das três portas das quais já falamos. Não tente, de forma alguma, mudar o valor contido neste registrador, pois isso pode causar um "crash" no hard do computador, levando a graves conseqüências no que diz respeito aos circuitos eletrônicos do seu micro.

Prendo-me, agora, a falar sobre os quatro bit's mais significativos do registrador "C", em especial dos bit's 4 e do bit 6.

O bit 4 do registrador "C" nos indica a condição da tecla de "CAPS LOCK". isto significa, que se a "caps" foi pressionada, haverá uma mudança do estado deste bit. Se o bit estiver resetado, significa que estarão disponível no teclado os caracteres maiúsculos, caso contrário, ou seja, se o bit 4 estiver setado, os caracteres minúsculos são os que estarão disponíveis. Isto nos será útil em especial nos micros EXPERT, visto que neste não existe um led de indicação do estado da "Caps Lock".

O bit 6 deste registrador da PPI nos dá a condição do motor do casssete. No caso do motor estar "ON", este bit estará resetado; caso o motor esteja "OFF", o bit estará setado.

Executando o programa você poderá testar as várias condições desses dois bit's. Digite-o, dê RUN, logo depois pressione a tecla de "Caps" e torne a dar RUN. Observou o que ocorreu com o bit 6?

Repita a operação para MOTOR ON e MOTOR OFF e observe o bit 4.

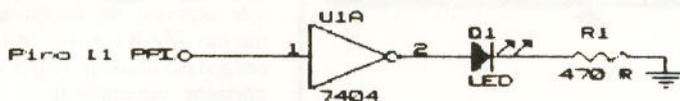
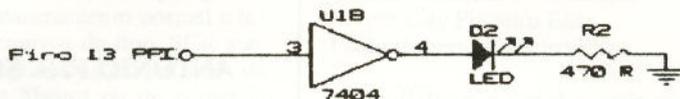
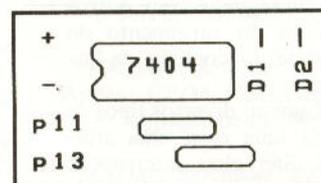
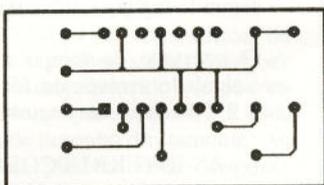
Baseado nisto, montamos um circuito que detecta a condição de "Caps Lock" e, aproveitando o embalo, implementamos também um led que nos indique a condição atual do motor do cassete.

O circuito em si pode ser observado na FIGURA 2 e um "lay out" da placa impressa com os componentes pode ser observada na FIGURA 3.

O que se propões é ligar ao pino 11 do integrado 8255, a PPI, um inversor lógico, e na saída do inversor colocarmos um led que nos dê a condição dos "caps". Desta forma, se a tecla "Caps" for pressionada e setada para maiúsculas, o led permanecerá aceso; caso contrário, ele ficará apagado. O mesmo procedimento pode ser implementado para o motor do cassete, ligando um inversor lógico no pino 13 da PPI. Assim, toda vez que o motor do cassete for acionado, o led acenderá; no contrário, o led estará apagado.

O circuito é composto de um integrado, o 74LSO4. Dentro dele se encontram disponíveis seis inversores lógicos, dos quais dois serão por nós usados. Este circuito necessita de uma alimentação de 5Vcc ligado ao seu pino 14, o que poderá ser conseguido dentro do próprio EXPERT, e o pino 7 deve ser levado ao terra. Os leds podem ser os comuns encontrados facilmente em qualquer casa do ramo com facilidade.

Quanto ao local onde seriam instalados os leds, deixo a cargo dos leitores a sugestão. Nos próximos números, na seção de cartas, procurarei divulgar os lugares de mais estética escolhidos pelos próprios. Assim, escrevam.

Pino 11 PPI — 1 — U1A 7404 — 2 — D1 LED — R1 470 R — LED de CAPS LOCK

Pino 13 PPI — 3 — U1B 7404 — 4 — D2 LED — R2 470 R — LED indicador de estado do MOTOR do gravador

```
1 ' REVISTA CPU No. 3 - JULHO 88
2 '
3 ' P.P.I.
4 '
5 ' CARLOS E. A. MOREIRA
6 '
10 CLS
20 A$=BIN$(INP(&HAA)):' LE PORTA C DA PPI
30 B$=RIGHT$("00000000"+A$,8)
40 FOR I=1 TO 4
50 PRINT "BIT ";8-I;" --> ";MID$(B$,I,1): PRINT
60 NEXT
```

Interrupções no Basic

ANTÔNIO F. S. SHALDERS

Um dos pontos mais fortes do MSX-BASIC é que este é capaz de manipular interrupções com um nível de interação com o programa que é liiteralmente impossível de ser conseguido com outros BASICs. Até antes do surgimento do MSX, isso era parte do domínio da linguagem de máquina.

O nosso MSX possui diversos tipos de interrupções, cada uma com uma utilização específica. São elas: interrupção por teclas, por tempo e por colisão de sprites. Os casos mais complexos são a por tempo e a de colisão.

AS INTERRUPÇÕES POR TECLAS

São as mais fáceis de serem usadas. Basta habilitarmos a tecla em questão que pode ser a barra de espaço (ou os disparadores dos joysticks), as teclas de funções ou quando as teclas CONTROL e STOP forem pressionadas simultaneamente. Feito isso, basta especificarmos para que linha será o desvio. Completado o ciclo, basta desabilitarmos a tecla em questão.

TECLAS DE FUNÇÃO

HABILITAÇÃO: KEY(N) ON
DESVIO: ON KEY GOSUB L1, L2, L3, ...,L10
DESABILITAÇÃO: KEY(N) OFF

BARRA DE ESPAÇO:

HABILITAÇÃO: STRIG(0) ON
DESVIO: ON STRIG GOSUB L
DASABILITAÇÃO: STRIG(0) OFF

OBS: STRIG(1) E STRIG(3): DISPARADORES DE J1
STRIG(2) E STRIG(4): DISPARADORES DE J2

TECLAS [CONTROL]+[STOP]

HABILITAÇÃO: STOP ON
DESVIO: ON STOP GOSUB L
DESABILITAÇÃO: STOP OFF

AS INTERRUPÇÕES POR TEMPO

São de grande utilidade quando se trata de programas de banco de dados ou processadores de texto.

Normalmente são usadas para prevenir uma eventual falta de energia elétrica durante a execução do programa, salvando automaticamente os arquivos de tempos em tempos.

Uma outra aplicação é a de abortar o programa no caso de uma senha não ser digitada em um dado intervalo de tempo.

DESVIO: ON INTERVAL=X GOSUB L
HABILITAÇÃO: INTERVAL ON
DESABILITAÇÃO: INTERVAL OFF

OBS: Neste caso a habilitação vem em segundo lugar.

O intervalo de tempo desejado pode ser calculado através da fórmula T * 60, onde T é o tempo em segundos.

AS INTERRUPÇÕES POR COLISÃO DE SPRITES

São de grande utilidade para quem se dedica à criação de pequenos jogos em BASIC. Com este tipo de interrupção, as colisões são facilmente detectáveis.

DESVIO: ON SPRITE GOSUB L
HABILITAÇÃO: SPRITE ON
DESABILITAÇÃO: SPRITE OFF

Em todos os exemplos dados, o L indica o número da linha para a qual o programa será desviado.

Uma boa maneira de se entender o funcionamento das interrupções é na prática. Para isso, recomendo a digitação e análise de funcionamento dos programas que acompanham o artigo.

No caso do que demonstra a colisão de sprites, um BEEP será emitido quando esta ocorrer.

```
100 '********************************
110 '*                              *
120 '*     INTERRUPCOES NO BASIC    *
130 '*                              *
140 '*  Antonio Fernando Shalders   *
150 '*                              *
160 '*         Revista CPU          *
170 '*                              *
180 '********************************
190 '
200 CLS:KEYOFF
210 '
220 'UTILIZACAO DAS TECLAS DE FUNCAO
230 '
240 KEY(3) ON
250 PRINT"PARA CONTINUAR, PRESSIONE (F
3)"
260 ON KEY GOSUB ,,280
270 GOTO 260
280 KEY(3) OFF:RETURN 320
290 '
300 'UTILIZACAO DE STRIG(0) (ESPACO)
310 '
320 STRIG(0) ON
330 CLS:PRINT"PARA CONTINUAR,PRESSIONE
(ESPACO)"
340 ON STRIG GOSUB 360
350 GOTO 340
360 STRIG(0) OFF:RETURN 400
370 '
380 'UTILIZACAO DE (CTRL)/(STOP)
390 '
400 STOP ON
410 CLS:PRINT"PARA CONTINUAR,PRESSIONE
(CTRL)/(STOP)"
420 ON STOP GOSUB 440
430 GOTO 420
440 STOP OFF:RETURN 480
450 '
460 'INTERRUPCAO POR TEMPO
470 '
480  CLS:PRINT"QUALQUER  TECLA PRESSIO
NADA DEPOIS  DE  10SEGUNDOS NAO SERA A
CEITA."
490 ON INTERVAL=600 GOSUB 540
500 INTERVAL ON
510 A$=INPUT$(1)
520 CLS:PRINT"UMA TECLA FOI PRESSIONADA"
530 INTERVAL OFF:PRINT:PRINT:END
540 PRINT"PASSARAM-SE MAIS DE 10 SEGUN
DOS"
550 RETURN 530
```

```
100 '*********************************
110 '*                               *
120 '*    INTERRUPCOES NO BASIC      *
130 '*                               *
140 '*  Antonio Fernando Shalders    *
150 '*                               *
160 '*          Revista CPU          *
170 '*                               *
180 '*********************************
190 '
200 ' INTERRUPCAO POR COLISAO DE SPRIT
ES 210 '
210 SCREEN 2,0
220 SPRITE$(1)=CHR$(0)+CHR$(255)+CHR$(0
)+CHR$(255)+CHR$(0)+CHR$(255)+CHR$(0)+C
HR$(255)
230 SPRITE$(2)=CHR$(255)+CHR$(129)+CHR$
(129)+CHR$(129)+CHR$(129)+CHR$(129)+CHR
$(129)+CHR$(255)
240 ON SPRITE GOSUB 300
250 FOR X=10 TO 245
260 SPRITE ON
270 PUTSPRITE 0,(X,96),,1
280 PUTSPRITE 1,(245-X,96),,2
290 NEXT X:GOTO 250
300 BEEP:SPRITE OFF:X=X+12:RETURN 290
```

Processadores

Quem uma vez já utilizou um processador de textos, certamente ficou achando a máquina de escrever uma coisa do passado, ultrapassada. Os processadores de texto oferecem recursos, principalmente de edição, que tornam a tarefa de datilografar um texto bem mais fácil, possibilitando, ainda, um trabalho sempre bem apresentado, pois pode-se mandar imprimir quantas vezes for necessário.

Para a linha IBM-PC podemos contar com inúmeros processadores e até sistemas que só trabalham em Winchester, ocupando um espaço de, aproximadamente, 4 MB (11 discos de face dupla) e que até permitem a alteração da disposição das teclas do teclado.

Tanto para a linha IBM-PC quanto para a linha MSX, existem, também, os Desktop Publishing, que reunem as facilidades dos processadores de texto e dos editores gráficos, permitindo a utilização simultânea de texto e gráficos ou fotografias e que serão analisados em outros artigos.

Podemos dizer que para a linha MSX existem três bons editores de textos, que são: Word Star, MSX Word e MSX Write

O Word Star trabalha em CP/M e necessita de 80 colunas, apesar de existir uma versão para 40 colunas mas que não funciona a contento. Sobre este editor existem inúmeros livros que poderão auxiliá-lo.

O MSX Write e o MSX Word são os mais utilizados pelos usuários do MSX, sendo que cada um possui características próprias.

Neste número da revista CPU aproveitaremos para falar sobre o MSX Write, fornecendo-lhe algumas dicas sobre o seu funcionamento.

Após o carregamento do programa, aguarde o Reset e chame o programa com CALL WRITE, pois este é um programa residente. Após ter chamado o programa, surgirá no vídeo o menú principal e você poderá começar a explorar todos os seus potenciais, lendo as intruções abaixo:

OPÇÃO E

Nesta opção, você entra no modo de edição. Na parte superior da tela aparece a opção que se está utilizando (EDIT), a memória livre, a indicação de WARP (justificação) e INSERT ligado.

INSERT

A indicação INSERT mostra que a tecla INS está na posição de inserção, permitindo ao usuário inserir um caracter, deslocando os demais, se houverem, para a direita. Caso a indicação seja O'TYPE, não haverá deslocamento dos caracteres para a direita, ou seja, o caracter sobre o qual se encontra o cursor será substituído pelo digitado.

TABULAÇÃO

A linha de tabulação permite ao usuário uma visão da posição do cursor.

As colunas com quadrado cheio marcam as tabulações, que podem ser modificadas pressionando-se a tecla CLS/HOME (c – retira tabulação/s – determina os pontos de tabulação).

TECLAS DE FUNÇÃO

F1 – Scroll

Pressionando-se F1, a parte superior da tela se modificará.

F1 : Movimenta o texto linha a linha
F2 : Texto para cima e para baixo
F3 : Texto para baixo rápido
F4 : B início do texto
F5 : E Fim de texto

F2 – Line

Nesta opção, pressionando-se BS (Back Space), você apaga o caracter à esquerda e com DEL o caracter à direira.

F3 – Word

Esta opção é semelhante à anterior, só que, em vez do caracter, é apagada a palavra.

F4 e F5 – Mark/b e Mark/e

Com Mark/b você marca o início de um bloco e com Mark/e o seu final.

Este bloco, parágrafo ou frase, pode ser copiado, mudado de posição, deletado, memorizado, retirado da memória e inserido no texto. Depois de delimitado, deve-se pressionar SELECT e F1(BLOCK). Se a escolha for copiar ou mover, o cursor deverá estar na posição desejada para a colocação do bloco.

SELECT

A tecla SELECT altera as teclas de função para Block, Find, Format e Ruler.

SELECT + F2 – FIND

A opção Find serve para procurar por uma palavra inserida no texto. O programa irá perguntar pela palavra a ser procurada e depois pedirá uma confirmação (C).

SELECT + F2 – CHANGE

Na opção CHANGE o procediimento será o mesmo ao do FIND, com a diferença de que a palavra será trocada por uma outra que o usuário tiver definido.

SELECT + F4 – FORMAT

A opção FORMAT modificará novamente as teclas de função para: PAGE, CENTER. LEFT e RIGHT.

CENTER – Centraliza a linha na qual se encontra o cursor.

LEFT – Arruma o parágrafo. ou frase, tantas colunas à esquerda quantas forem determinadas, inserindo, para isto, quantos espaços forem necessários.

RIGHT – O mesmo do anterior com a diferenç de que o texto será arrumado (justificado) à direita.

SELECT + F4 – WARP

A opção WARP dispõe as palavras na linha na qual se encontra o cursor sem dividí-las, podendo ser ativada, ou desativada, quando se desejar. Uma indicação de que a função WARP está ativada é dada pelo programa na parte superior direita da tela. Portanto, a função WARP deve ser utilizada sempre que desejarmos efetuar a justificação ou alinhamento do texto.

P – IMPRESSÃO

No menú principal, pressione a tecla "P".

O programa irá perguntar se o arquivo a ser impresso é uma continuação de uma impressão anterior, caso esta tenha existido. Se você responder a esta pregunta com "Y" (Yes), a numeração das páginas será a continuação da impressão anterior.

Nesta opção são feitas várias perguntas, a fim de que o usuário possa definir por completo o formato de impressão.

F – GERENCIAMENTO DE ARQUIVOS

Nesta opção você pode ler, gravar, apagar, pedir o diretório ou formatar um disco. As opções de leitura ou gravação perguntam se o periférico a ser utilizado é o drive ou o cassete.

T – IMPRIMIR DIRETO NA FOLHA

Através desta opção você poderá utilizar o seu processador de texto como se fosse uma máquina de escrever, ou seja, tudo o que for digitado será impresso, após ser dado um CR (retorno de carro).

Tratamento de erros

ANTÔNIO F. S. SHALDERS

Embora não pareça, o tratamento de erros no MSX-BASIC é uma tarefa muito simples de ser feita, pois este conta com instruções poderosíssimas para a detecção e manipulação destes.

Estas funções quase nunca são usadas pela grande maioria dos usuários da linha MSX, a não ser por um pequeno número que usa o micro na área científica ou outra aplicação profissional.

O seu MSX tem normalmente disponíveis nada menos que 59 mensagens de erro. Este número pode subir até 255, pois o usuário pode definir seus próprios erros, desde que os códigos destes estejam compreendidos entre 60 e 254. O de número 255 pode ser usado por algum periférico e em geral não é usado pelo usuário (quem irá definir mais que 194 erros???).

A DETECÇÃO DOS ERROS

Para detectarmos se houve ou não um erro durante a execução do programa e se nãoq queremos que o interpretador BASIC pare, devemos fazer uso do desvio por erro, através da instrução ON ERROR GOTO.

Esta instrução faz com que o programa seja desviado da rotina de tratamento ou contornação do erro. O tratamento de erros é a técnica de modificar os dados para que o programa prossiga corretamente a sua execução. O contorno apenas faz com que o programa não seja interrompido pelo interpretador.

Podemos descobrir facilmente qual o erro ocorrido, bastando usar, para isso, a função ERR, que retorna o código do erro detectado. O número da linha onde o erro ocorreu também é facilmente obtido, usando-se a função ERL.

O PROSSEGUIMENTO DO PROGRAMA

A não interrupção do programa é conseguida através da instrução RESUME. O formato desta é RESUME XXX, onde XXX é o número da linha de re-ínicio.

Se for desejado o prosseguimento na linha seguinte à que ocorreu o erro, basta usarmos o formato RESUME NEXT.

INTERRUPÇÃO PROPOSITAL DO PROGRAMA

Isto é possível de ser feito pelo próprio usuário, pois, como já disse, podemos definir seus próprios erros através da instrução ERROR XXX onde XXX é o código do erro desejado. Se usado um código existente, o interpretador pensará que um erro comum aconteceu e, caso o programa tenha uma instrução de desvio para erros, executá-la-á. Caso contrário, a execução do programa será interronpida.

Caso o número do erro seja superior a 59, o usuário deverá tratá-lo convenientemente através de ERR e ERL. Uma aplicação prática é para testarmos senhas de acesso de um determinado programa.

Em qualquer caso é bom não esquecermos de usar o RESUME.

CONCLUSÃO

As rotinas de manipulação e tratamento dos erros no MSX são recursos importantíssimos na programação profissional. Convém lembrar que o BASIC é uma das pouquíssimas linguagens que possuem esta facilidade.

Um programa em BASIC, se bem estruturado e usando todos os recursos do MSX-BASIC, pode ser muito melhor que programas relativamente bem estruturados em Pascal, por exemplo.

Em programas que lidam com cálculos e em bancos de dados, tais recursos são indispensáveis, pois tornam o programa muito mais seguro.

Analise o programa que acompanha este artigo e veja você mesmo as potencialidades que uma manipulação correta de erros pode proporcionar.

```
200 ON ERROR GOTO 320
210.WIDTH 40:KEY OFF:CLS
220 INPUT"Senha:";S$:PRINT
230 IF S$()"MSX" THEN 240 ELSE 250
240 ERROR 60      :'ERRO DEFINIDO
250 A=SQR(-1)     :'ERRO MATEMÁTICO
260 A/4           :'ERRO DE SINTAXE
270 DIM A(10)
280 DIM A(20)     :'DIM REDEFINIDO
290 A="T"         :'TIPO DESIGUAL
300 LOAD"XXX"     :'ARQUIVO NÃO EXISTE
310 PRINT:PRINT"ERROS DEVIDAMENTE MANIPU
LADOS !":PRINT:PRINT:LIST 220-300
320 E=ERR:L=ERL
330 IF E=5 THEN PRINT "ERRO MATEMÁTICO N
A LINHA";L:RESUME NEXT
340 IF E=2 THEN PRINT "ERRO DE SINTAXE N
A LINHA";L:RESUME NEXT
350 IF E=10 THEN PRINT "REDEFINIÇÃO DE M
ATRIZ NA LINHA";L:RESUME NEXT
360 IF E=13 THEN PRINT "TIPO ERRADO DE V
ARIAVEL NA LINHA";L:RESUME NEXT
370 IF E=53 THEN PRINT "ARQUIVO NÃO EXIS
TE NO DISCO NA LINHA";L:RESUME NEXT
380 IF E=60 THEN PRINT "ERRO DE SENHA NA
 LINHA";L:RESUME NEXT
```

Graphic Master - análise

MSX INFORMÁTICA

Existem no mercado inúmeros Editores de Desenho para a linha MSX, cada um com alguma característica específica que atende determinada necessidade do usuário. Uns têm como ponto forte letras especiais, outros facilidades de edição de telas. O difícil é encontrarmos um editor que atenda, pelo menos, a maioria das nossas expectativas. Executar traços contínuos, linhas, desenhar quadrados e circulos, colorir, apagar, é o que, de modo geral, todos os editores de desenho fazem. Nossos problemas começam quando queremos trabalhar simultaneamente com letras, "shapes" (desenhos prontos), simetrias ou cores intermediárias, por exemplo. Isto somente era possível através de várias etapas de trabalho e de uma incrível ginástica (nem sempre bem sucedida) para converter nossa tela do formato de um editor de desenho para outro.

Diante dessas limitações e da ausência do programa ideal dos nossos sonhos (que é sempre diferente para cada um), o programa ideal é aquele que possui o maior número de recursos disponíveis, além de um kit de ferramentas de desenho que possibilite criar e armazenar diferentes elementos para uso imediato ou posterior. E é isto que o Graphic Master, que é comercializado pela MSX Informática, oferece ao usuário do MSX: recursos básicos e alternativos, além de ferramentas para criação. Vejamos mais detalhadamente o que torna o Graphix Master altamente atrativo.

Um menu principal com 22 opções e vários outros secundários permitem compor inúmeras alternativas de desenho. Além disso, possui duas telas diferenciadas de edição e quatro quadros de "shapes" variados. Seu funcionamento básico, de extrema simplicidade, facilita sua utilização: através das teclas F1 e F2 são ativadas e desativadas as opções existentes. As teclas cursoras servem para desenhar, aceitando também o mouse na execução de desenhos. Embora disponível apenas em cartucho de 32 K, permite a gravação das telas criadas tanto em disco como em fita, bem como a sua impressão através do ASMCOCAR ou do Printing Pack.

Possui todas as funções comuns aos editores de desenho: traços contínuos, retas, pontilhados, elipses e círculos. Este último permitindo o traçado de círculos fechados, abertos ou semi-círculos. Reproduz letras e símbolos do teclado em caracteres cheios e em qualquer cor. Tem borracha, lupa de aproximação e função de cópia exata ou invertida de uma área de tela definida pelo usuário. Executa movimento de rotação lateral e apagamento. Seus pontos fortes são as opções de traços e "patterns", de espelhos, de uso de cores, de "shapes" prontos, de arquivamento de "shapes" próprios, e de edição. Vejamos estes aspectos mais detalhadamente.

TRAÇOS E "PATTERNS"

Existem 8 tipos diferentes de traçados, cada um com 3 espessuras ou concentração de pontos diferentes, perfazendo um total de 24 opções que poderão ser utilizadas quando do uso de linha ou traço contínuo. Estas opções permitem trabalhar com formas mais arrendondadas ou criar a ilusão de esfumaçamento, evitando aquela sensação de desenho feito "à régua". Da criatividade do desenhista e do uso destes recursos podem surgir outros inúmeros efeitos. Neste mesmo menu se pode optar entre 10 modelos de "patterns" que podem ser usados para preenchimento de campos de molduras em qualquer cor. A combinação de um determinado "pattern" e do tipo de traço aumenta as possibilidades de criação, pois os efeitos serão distintos conforme a combinação escolhida. O uso desses "patterns" em molduras em conjunto com alguns dos "shapes" já existentes e de letras, permitem a elaboração de cartões com bastante efeito.

ESPELHOS

Neste menu, o usuário, além das opções de cópia invertida em ambos os sentidos, de uma parte de um desenho na tela, conta com três opções de espelhamento. Ou seja, pode reproduzir simultaneamente um traço ou forma tanto no sentido vertical ou horizontal, como ainda nos quatro cantos da tela. O posicionamento da figura "espelho" dependerá do ponto de origem em que iniciar o traçado. Este recurso evita o minucioso trabalho de cópia e inversão do desenho para outro ponto da tela, além de dispensar cálculos para posicionamento simétrico da cópia em relação ao original. Isto, é claro, tem como resultado um trabalho limpo e uma preceisão nem sempre obtida quando outros meios são utilizados.

CORES

As opções de cores, tanto para traçado como para pintura são bastante ricas no Graphic Master. Isto se deve a duas possibilidades internas dos menus de cores: a primeira é a possibilidade de se mudar a cor de um campo ou traço já pronto para outra cor, automaticamente. Isto é extremamente útil quando, com um desenho já pronto, percebemos que outra cor ficaria melhor no conjunto. Basta colocar o cursor no campo ou no traço a ser alterado, escolhermos a opção de trasferência de cor e indicarmos a velha e a nova cor a ser utilizada. A velha cor é substituída imediatamente, sem prejuízo do restante do desenho e das outras cores existentes. Outro recurso de cor interessante é a possibilidade de se criar cores intermediárias, definindo duas cores diferentes para serem misturadas. Embora esta combinação resulte na produção de um campo hachuriado envolvendo as duas cores originais, uma escolha de cores com certo critério faz com que o hachuriado praticamente passe desapercebido, criando a ilusão de uma única cor de tonalidade secundária no desenho final. Isto aumenta enormemente as possibilidades de se obter efeitos alternativos no desenho.

"SHAPES"

O Graphic Master é o único editor de desenho para MSX que já tem incorporado ao programa um rol de desenhos prontos para serem utilizados. Trata-se de um campo definido como ROM, na verdade composto de vários quadros que o usuário pode acessar teclando F1 e que vão apresentar as figuras disponíveis. Existem figuras humanas, construções, flores, veículos, etc, todos coloridos, que podem ser transportados diretamente para a tela de desenho. Caso o usuário deseje alterar as figuras ou a sua cor de fundo, poderá fazê-lo através da tela de edição. Existe, ainda, a possibilidade de se criar "shapes" próprios ou letras através de edição. Para facilitar o uso destes novos "shapes" durante a execução do desenho, o quadro ROM pode ser transformado em RAM, onde as novas formas criadas permanecerão arquivadas. Isto permite o uso gra-

dual das mesmas enquanto se cria uma tela. Claro que criar novos "shapes" para perdê-los ao desligar o micro não teria vantagem nenhuma. Por isso o Graphic Master tem uma opção de gravação dos shapes presentes no quadro RAM, em um arquivo tipo .PRT que, quando carregado posteriormente, é diretamente inserido novamente no quadro RAM para facilidade do usuário. (Este modo de armazenamento é exclusivo para manipulação de "shapes". O armazenamento normal e telas é feito em arquivos do tipo .SCR que, depois, podem ser carregados através do próprio Graphic Master ou do comando BLOAD"TELA.SCR",S.)

EDIÇÃO

Existem neste editor duas alternativas de edição de desenhos, além da comodidade do desenho já poder ser criada a cores:

EDIÇÃO PARA CRIAÇÃO

Seguindo os moldes de um editor de sprites, nesta tela podemos criar qualquer letra ou desenho multicolorido que poderá ser transportado ou para o quadro RAM, ou para a tela de desenho diretamente. As opções inversas também estão disponíveis: transportar um desenho da tela ou do quadro RAM para a tela de edição para ser retrabalhado. Também é aí que se encontra a opção de gravação de arquivos .PRT (de "shapes").

EDIÇÃO PARA ALTERAÇÃO

Esta opção remete para um editor similar ao anterior, com o mesmo esquema de funcionamento. A diferença básica é quem neste, desenho ou porção do mesmo a ser editada, é traduzido diretamente da tela maior para a de edição e, a medida em que a alteração vai sendo feita, ela já é reproduzida no desenho principal. Esta tela de edição tem como opção, além das cores, a movimentação para cima, para baixo e nas laterais do desenho exposto na tela de edição, permitindo que se "caminhe" no desenho maior do vídeo sem necessisade de voltar a ele para "buscar" outro pedaço de desenho para ser editado. Isto acelera e facilita a operação.

Embora todos os recursos existentes no Graphic Master sejam de extrema utilidade, sem dúvida são os recursos de edição e a disponibilidade de "shapes" que tem maior peso na decisão de compra do cliente.

Outra vantagem importante de ser mencionada é a de que o programa possui uma rotina interna de controle de erros. isto evita aqueles inconvenientes de outros editores em que se pode perder totalmente uma tela já desenhada devido a algum problema de carregamento ou gravação ou por falta de espaço no disco, por exemplo. No Graphic Master isto não acontece. A mensagem de ERRO é apresentada e basta retornar à opção anterior para termos chance de trocar o disco sem perder a tela.

LIVROS

dBase II Plus
Ciência Moderna Computação
Sérgio Guy Pinheiro Elias
Paulo Roberto Pinheiro Elias

O dDBase é um dos mais poderosos sistemas de gerenciamento de banco de dados e, através desta publicação, o leitor terá acesso a todo o potencial que este programa apresenta, mesmo que nunca tenha operado com este tipo de software, pois o livro é dirigido aqueles que nunca montaram um banco de dados.

No primeiro e segundo capítulo são dadas informações gerais do uso do sistema operacional (format, back-up, etc) e de como instalar e rodar o dBase. O programa utilizado pelos autores é o dBase II Plus fornecido pela Datalógica.

No terceiro capítulo são apresentados todos os comandos, agrupados por tópicos, como por exemplo: criação de um arquivo, manutenção, visualização, etc.

Todos os comandos são apresentados neste capítulo, e, sempre que necessário, os autores utilizam exemplos práticos, facilitando, assim, a assimilação dos conceitos por parte do leitor.

O quarto capítulo é reservado à programação em dBase II, que irá fornecer ao leitor condições para uma total e correta manipulação das informações armazenadas.

No último capítulo é feita uma descrição dos quatro utilitários que acompanham o dBase II Plus e que permitem a geração automática de programas básicos para a manipulação de um banco de dados.

No apêndice são fornecidos 2 programas em linguagem dBase que fazem uso de recursos do dBase II Plus.

PROGRAMAÇÃO PROFISSIONAL EM BASIC
Roberto Nassau Watanabe
Editora Aleph

A programação é uma arte e, como tal, requer uma dedicação especial. O programador que se dedicar a conhecer profundamente a maneira como um programa é executado descobrirá uma série de "macetes" que tornam o programa mais eficiente.

Programação profissional em Basic tem por objetivo apresentar os tópicos que permitirão aos programadores em vias de profissionalização estruturar um esquema de produção de programas de uma maneira rápida e racionalizada.

O livro encontra-se dividido em 9 capítulos que são:

1 – Introdução
2 – Programação
3 – Técnicas de entrada de dados
4 – Acesso de discos
5 – Técnicas de saída
6 – Manipulação de variáveis
7 – Desenvolvimento de um sistema
8 – Eficiência de um sistema

E dois apêndices:

– Conversão de comandos (entre Apple – CP/M, MSX e IBM/PC)
– Ocupação de memória

Como conclusão do livro, o leitor obtém um programa para efetuar o controle de estoques, cujos módulos são apresentados, desenvolvidos e explicados no decorrer do livro.

MSX COM DISC DRIVE
Nelson Casari
Editora McGraw-Hill

Guardar todos os comandos e, principalmente, seus recursos adicionais do sistema operacional, não é uma tarefa muito fácil, mesmo para aquele usuário que utiliza com freqüência o sistema.

Em MSX COM DISK DRIVE foram reunidos todos os recursos operacionais (MSX Disk Basic, MSX Dos e HB-MCP) de forma a proporcionar aos leitores um manual de consultas.

Os primeiros capítulos facilitam a aquisição de informações básicas sobre: memórias auxiliares, Disk Drives ou processadores de disco, disquete ou mini discos, interfaces, sistemas operacionais padrão MSX, instalação e ativação dos sistemas, operando com um ou dois disk drives.

Nos capítulos 9, 10 e 11 são apresentados todos os comandos, instruções e funções do MSX Basic, MSX Dos e do HB-MCP, respectivamente, sendo fornecidos exemplos práticos e mensagens, além da análise dos possíveis casos de erro.

Os apêndices, de extrema importância, são ao todo 9, onde destacamos as tabelas de mensagens e tabela de funções, além de um programa gerenciador de arquivo sequencial e um outro gerenciador de arquivo aleatório.

Função do 1º Grau

GUILHERME A. L. DA SILVA

A função do 1º grau é do tipo:
F(x)=Ax + B

Tendo como condição que o "A" seja maior ou menor que zero e pertencente ao conjunto dos Reais. "B" pode ser qualquer número real, determinando o ponto em que a reta irá cruzar o eixo Y.

O PROGRAMA

O programa pede, inicialmente, os valores de "A" e "B" e, a seguir, pede a confirmação dos valores fornecidos, fornecendo:

– o coeficiente (angular ou linear);
– o valor da ordenada (0,Y);
– o zero da função, ou seja, a abcissa (X,0);
– o estudo do sinal.

Após ter fornecido estes dados, novos dados são solicitados para traçar o gráfico, sendo que estes servem apenas para melhorar a apresentação do gráfico.

O gráfico é, então, mostrado na tela, podendo também ser impresso, bastando pressionar a tecla ESC.

VARIÁVEIS DO PROGRAMA

FN, PX(A,B,X)	= resolução da equação para achar os pontos
A	= termo "A"
B	= termo "B"
X	= icógnita – contém a resolução da equação
ROT	= seleciona a rotina
A$	= string de A
B$	= string de B
H	= valor de A para uso no estudo do sinal
AA	= valor de A para uso no gráfico
BB	= valo de B para uso no gráfico
V	= arredondamento de X
PX( )	= matriz dos quatro pontos extras no eixo X
PY( )	= matriz dos quatro pontos extras no eixo Y
CY	= coordenada central Y
CX	= coordenada central X

```
1 REM ---------- FUNCOES DO 1
GRAU
2 REM ---------- GUILHERME A.L
. DA SILVA
3 REM ---------- PARA LINHA MS
X
4 REM ---------- 22/06/88
5 REM ---------- GUARARAPES-SP
10  CLEAR 10000
20 OPEN"GRP:"FOR OUTPUT AS #1
30 DEF FN PX(A,B,X)=A*X+B
40 ' PROG. FUNCAO
50 ROT=1
60 COLOR 15
70 CLS:KEY OFF
71 COLOR 5,1
80 PRINT:PRINT"    EXAMINADOR
 DE FUNCOES DO 1 GRAU
110 ON ROT GOSUB 130,280,370,
460,680,780,900
120 GOTO 70
130 ' F(X) NA TELA
140 LOCATE15,4:PRINT"Funcao"
150 LOCATE 6,8:INPUT"A=";A$:A
=1
160 LOCATE24,8:INPUT"B=";B$:B
=0
170 A=VAL(A$):B=VAL(B$):C=VAL
(C$)
180 H=A:BB=B:AA=A
200 IF A=0 THEN RUN
210 IF A$="-1" THEN A$="-"
220 IF A$="1" THEN A$=""
230 IF B$="0" THEN B$=""
240 IF B$>"1" THEN B$="+"+B$
250 LOCATE10,12:PRINT"F(X)= "
;A$;"X";B$
260 LOCATE 13,16:INPUT"CONFIR
MA";A$:IF A$="N" THEN RUN
270 ROT=2:RETURN
280 ' A DE F(X)
290 COLOR 15,4
300 LOCATE 11,4:PRINT"ANALISE
 DE A"
310 IF A>0 THEN Z$="crecente"
:W$="positiva"
320 IF A<0 THEN Z$="decrescen
te":W$="negativa"
330 LOCATE 8,10:PRINT"Funcao
"+Z$
340 LOCATE 8,11:PRINT"Inclina
cao "+W$
350 LOCATE 12,16:PRINT"<RETUR
N>":IF INKEY$="" THEN 350
360 ROT=3:RETURN
370 ' F(X) DE B
380 COLOR 12,1
390 LOCATE 11,4:PRINT"ANALISE
 DE B"
400 IF B=0 THEN B$="0":Q$="li
near"
410 IF B<>0 THEN Q$="angular"
420 LOCATE 8,10:PRINT"Coefici
ente "+Q$
430 LOCATE 8,11:PRINT"O 1 pon
to e: (0,";B$;")"
440 LOCATE 13,16:PRINT"<RETUR
N>":IF INKEY$="" THEN 440
450 ROT=4:RETURN
460 ' O da f(x)
470 COLOR 15,8
471 IF B=0 THEN B$=""
480 LOCATE 10,4:PRINT"ZERO DA
 FUNCAO"
```

```
490 LOCATE12,6:PRINT"F(X)= ";
A$;"X";B$
500 LOCATE12,7:PRINT"0=";A$;"
X";B$
510 LOCATE12,8:PRINTA$;"X";B$
;"=0"
511 IF B=0 THEN X=0:X$="0":LO
CATE12,8:PRINT"X=0";SPC(5):GO
TO 650
520 IF B<0 THEN B$="+"+MID$(B
$,2,1) ELSE B$="-"+MID$(B$,2,
1)
530 B=B*(-1)
540 LOCATE12,9:PRINTA$;"X=";B
$
560 IF A=1 THEN 650
581 IF A<0 THEN A$=STR$(A)
600 LOCATE12,10:PRINT"X=";STR
$(B);"/";STR$(A)
610 X=B/A
620 V=INT(1000*X)/1000
630 X$=STR$(V)
640 LOCATE12,11:PRINT;"X="+X$
650 LOCATE 10,14:PRINT"O 2 po
nto e: (";X$;",0)"
660 LOCATE 12,16:PRINT"<RETUR
N>":IF INKEY$="" THEN GOTO 66
0
670 ROT=5:RETURN
680 ' SINAL
690 COLOR 7,1
700 LOCATE 9,4:PRINT"REGRA DE
 SINAL DA F(X)"
710 LOCATE10,8:PRINT"Qdo Y=0
entao X=";X$
720 IF SGN(H)=-1 THEN Q$="<":
W$=">":GOTO 740
730 IF SGN(H)=1 THEN Q$=">":W
$="<"
740 LOCATE10,9:PRINT"Qdo Y>0
entao X";Q$;X$
750 LOCATE10,10:PRINT"Qdo Y<0
 entao X";W$;X$
760 LOCATE 14,16:PRINT"<RETUR
N>":IF INKEY$="" THEN 760
770 ROT=6:RETURN
780 ' achando x,y
790 COLOR 15,13
800 LOCATE 12,4:PRINT"ACHANDO
 (X,Y)"
810 FOR I=1TO4
820 LOCATE 2,7+I:INPUT"X:";PX
(I)
830 PY(I)=FN PX(AA,BB,PX(I))
840 IF PX(I)>13 OR PY(I)<-11T
HEN GOTO 820
850 IF PY(I)>8OR PY(I)<-9THEN
 GOTO 820
860 LOCATE 13,7+I:PRINT"Y:";P
Y(I)
870 LOCATE 23,7+I:PRINT"Ponto
 (";PX(I);",";PY(I);")"
880 NEXT
890 LOCATE13,17:PRINT"<RETURN
>":IF INKEY$="" THEN 890
900 ' GRAFICO
910 COLOR 15,1,4:SCREEN2
920 CY=93:CX=130
930 FOR H=13TO 190 STEP 10
940 FOR G=0 TO 250 STEP 10
950 PSET(G,H),6
960 NEXT G,H
970 LINE  (0,0)-(250,13),4,BF
980 COLOR 15
990 FORI=1TO4:PY(I)=CY+((PY(I
)*-1)*10):PX(I)=CX+(PX(I)*10)
:NEXT
1000 BB=(BB*-1)*10:X=X*10
1010 PRESET (61,4):PRINT#1,"E
sboco do grafico"
1020 PRESET (62,4):PRINT#1,"E
sboco do grafico"
1030 LINE(0,13)-(250,190),15,
B
1040 LINE(0,CY)-(250,CY)
1050 LINE(CX,13)-(CX,190)
1060 FOR I=0 TO 250 STEP 10
1070 LINE(I,CY+2)-(I,CY-2)
1080 NEXT
1090 FOR I=13 TO 190 STEP 10
1100 LINE(CX+2,I)-(CX-2,I)
1110 NEXT
1120 LINE  (PX(1),PY(1))-(PX(
2),PY(2)),15
1130 LINE -(PX(3),PY(3)),15
1140 LINE  -(PX(4),PY(4)),15
1150 LINE (CX+X,CY)-(CX,BB+CY
),15
1190 FOR I=1TO4:CIRCLE(PX(I),
PY(I)),2,8:NEXT
1200 FOR I=1TO4:PAINT(PX(I),P
Y(I)),8:NEXT
1210 CIRCLE(CX+X,CY),2,12:PAI
NT(CX+X,CY),12
1220 CIRCLE(CX,BB+CY),2,12:PA
INT(CX,CY+BB),12
1230 R$=INKEY$:IF R$="" THEN
1230
1240 IF R$=CHR$(27) THEN 1500
1250 CLS:KEY ON:END
1299 REM
1500 REM ---- ROTINA DE IMPRE
SSAO DA
1501 REM ---- AGUIA INFORMATI
CA
1505 POKE&HF417,1
1510 DEFINTA-Z:FOREN=0TO5:REA
DCO:POKE&HE000+EN,CO:NEXT
1520 DATA 62,0,205,165,0,201:
DEFUSR=&HE000
1530 F=PEEK(&HF3EA)
1540 LPRINTCHR$(27);"A";CHR$(
1)
1550 FORXX=0TO255
1560 LPRINTCHR$(27);"K";CHR$(
0);CHR$(2);
1570 FORYY=191TO0STEP-1
1580 IF POINT(XX,YY)=FTHEN160
0
1590 POKE&HE001,1:GOTO1610
1600 POKE&HE001,0
1610 G=USR(0)
1620 NEXTYY:LPRINTCHR$(10);:N
EXTXX
```

MÁXIMAS E MÍNIMAS

J. L. FONSECA

Vamos, hoje, falar sobre um assunto que está muito em voga entre os programadores e que se chama software re-usável.

A idéia de re-usar rotinas e pedaços de código desenvolvidos anteriormente não é nova nem original, mas é, no entanto, muito atraente e os fatores que implicam nisso são, basicamente, a economia e a segurança.

A economia é causada pelo fato de que não é necessário perder tempo recriando rotinas de uso corriqueiro, bastando apenas retirá-las de uma biblioteca e incluí-las no nosso programa, o que quase todas as linguagens para o MSX permitem com facilidade, eliminado o tempo e o esforço de digitá-las novamente, o que poderia, inclusive, causar erros díficeis de encontrar. A segurança se deve ao fato de que se estas rotinas funcionaram bem em outros programas, deverão fazê-lo também aqui, eliminando o tempo gasto em depuração das mesmas no programa atual.

Para que nós possamos desfrutar destas vantagens é necessário, no entanto, seguir algumas normas durante o desenvolvimento dos nossos programas. Estas normas são ditadas pelo bom senso e pelo estilo individual de cada programador. Daremos, a seguir, algumas normas ditadas pelo bom senso e, obviamente, pelo nosso estilo pessoal.

A primeira norma é que as rotinas por nós desenvolvidas devem ser modulares em todos os sentidos, isto é, as variáveis usadas pela rotina devem ser criadas dentro da rotina e, se possível, ser do conhecimento apenas da mesma, e o código desta deve estar contido em um ou mais blocos perfeitamente definidos. Esta norma, como podemos ver, vem ao encontro da idéia de programação estruturada que foi vista em número anterior e, na verdade, uma reforça a validade da outra.

A segunda norma refere-se à documentação que, se é importante num programa, é infinitamente mais importante numa rotina de bilbioteca. Todas as rotinas da nossa bilbioteca devem ter como cabeçalho um grupo de comentários dizendo qual a sua função, como deve ser chamada, quais as variáveis externas que por ventura precise acessar, e se as mesmas fazem parte da biblioteca, limitações conhecidas e, finalmente, o nome do autor e a data da implementação.

A terceira norma é a ausência de efeitos colaterais, isto é, a rotina não deve modiificar nada no programa principal além do que ela se propõe explicitamente a alterar, pois, de outro modo, poderia causar defeitos dificílimos de encontrar. Esta norma é outra que está dentro da idéia da programação estruturada, mas que deve ser muito mais enfatizada neste contexto.

Uma outra norma é a robustez, ou seja, a rotina deve, dentro do possível, prever e tratar todos os erros dentro do seu corpo e retornar resultados coerentes, mesmo quendo receber dados errados na sua entrada, e, caso seja necessário, deve passar um sinal de erro ao programa principal para que este tome as providências necessárias.

Muitas outras normas e regras poderiam ser citadas, mas estas são a base de uma boa bilbioteca saudável e bem comportada que muito auxiliará o seu possuidor na hora de desenvolver novos programas. E lembre-se que o bom programador é, antes de tudo, um colecionador de boas rotinas e o feliz possuidor de uma ampla biblioteca das mesmas.

Caso você tenha rotinas curtas e úteis que gostaria de compartilhar com todos os leitores, pode mandá-las para esta seção que as mesmas serão publicadas sempre que as limitações de espaço o permitirem. Serão também bem vindas sugestões sobre os assuntos que gostariam de ver tratados nesta coluna.

Linguagens no MSX

ANTÔNIO F. S. SHALDERS

LINGUAGENS PARA O MSX

Às vezes sou indagado sobre quais linguagens existem para o MSX, e quais os prós e contras de cada uma. Embora não pareça, a pergunta não é fácil de ser respondida, pois, além da infinidade de linguagens que o MSX é capaz de rodar, isso é muito impessoal.

Tentarei, pois, transmitir-lhes algo sobre o assunto, que é por demais fascinante.

COMO CONSEGUIR OS COMPILADORES:

Neste aspecto, os possuidores da interface de disco da SHARP HB-3600 levam uma enorme vantagem, pois têm acesso ao vastíssimo universo do sistema operacional CP/M, sem dúvida alguma o sistema operacional mais difundido do Planeta. Existem milhares de programas escritos para este sistema, entre eles alguns dos melhores compiladores!

Outro aspecto importante é que o utilitário DSKCNV.COM, que o acompanha, permite o acesso às bibliotecas do CP-500, S-600, S-700, Itautec I-7000 e outros.

Com um pouco mais de trabalho, podemos fazer conversões do EBC-4020 para o MSX, bastando converter em duas etapas: primeiro de EBC para I-7000 e, depois, para MSX. Para quem não conhece, o EBC-4020 é um dos micros do tipo CP/M mais rápidos e versáteis que existem. Pena que estão caindo em desuso.

Para os não possuidores da HB-3600 ou similares, a situação é menos favorável, mas, mesmo assim, há um bom número de compiladores para o MSX-DOS.

Já para os que possuem apenas a unidade cassete restam poucas alternativas, mas uma delas é fantástica: o compilador Pascal HISOFT.

Este compilador é um dos mais rápidos desta linguagem existente para Z-80. Os usuários do ZX-ESPECTRUM que o digam!

A seguir, dou uma lista dos compiladores mais facilmente encontráveis, assim como a configuração necessária para utilizá-los.

MSX-DOS:	TURBO PASCAL 3.00
	muLISP 80/83
	AZTEC C
	NEVADA FORTRAN 3.0
	BDS C
	LOGO
	MICRO-PROLOG
CP/M:	IDEM MAIS:
	muSIMP/muMATH 80/83
	MICROSOFT FORTRAN
	MICROSOFT COBOL
	MICROSOFT BASIC
	BASCOM
	MVF-FORTH
CASSETE:	HISOFT PASCAL
	COMP-32 (BASIC)
	LOGO (VERSÃO EM BASIC)

O que proponho é abordarmos um problema específico através de várias linguagens diferentes, a fim de que o leitor possa escolher a mais versátil para solucionar tal problema, a que mais lhe convier ou, simplesmente, a que ele mais simpatizar.

O teste será feito no clássico fatorial. Embora isso já esteja muito batido, é um problema fácil de ser entendido por leigos em matemática e envolve um grande número de operações aritméticas.

Obviamente, este teste serve para testarmos se uma linguagem é adequada ou não para cálculos.

O fatorial de um número é dado pela série numérica F = N * (N-1) * (N-2) * ... * 1, onde F é o fatorial do número e N o número em questão, que deve ser obrigatoriamente positivo e inteiro.

O fatorial de 6, por exemplo é igual a 720, pois 6 * 5 * 4 * 3 * 2 * 1 = 720.

São dadas, a seguir, algumas listagens do programa de fatorial em BASIC, FORTRAN, C, APL, LISP, Pascal, Algol, Forth e Logo.

As mais rápidas são o C e o Forth, pois estas são de médio nível. A de alto nível mais rápida é o FORTRAN. Note que o expoente no Nevada FORTRAN 3.0 é de 3 dígitos!. Já LISP trabalha por um processo diferente, o que lhe confere precisão infinita, dependendo apenas da memória do micro! No caso do muSimp/muMath, que é uma linguagem para cálculos pesados, você pode pedir tranquilamente o fatorial de 10000, que ele responderá com TODOS os algarismos!

muLISP, muSIMP/muMATH e Logo são muito semelhantes no geral. Logo pode ser considerado uma variação do LISP. Já o muSIMP/muMATH é um LISP superimplementado para cálculos de qualquer tipo e precisão, além de trabalhar com matemática simbólica!

C, FORTH, Pascal, Algol e Basic são de uso mais geral.

Forth, apesar de ser uma linguagem bem estranha (se comparada com as tradicionais) pode ser usada em praticamente todos os ramos. Com C e Algol até sistemas operacionais podem ser escritos!

O Pascal, o Algol e o Forth são linguagens estruturadas e procedurais. O Pascal é de fácil aprendizado e de desempenho muito bom, mas o raciocínio usado na elaboração de programas é bem diferente do BASIC, pois obriga o programador a estruturar todo o programa antes de digitá-lo, o que não acontece com o BASIC.

Embora o BASIC seja uma linguagem lenta, por sua complexidade e pelo fato de ser interpretado em 90% dos casos, possui um dos sets de instruções e comandos mais completos que existem.

Aliás, o MSX-BASIC é uma das versões BASIC mais poderosas existentes e é, sem dúvida, a melhor para máquinas de 8 bits. Se você deseja saber qual é o melhor BASIC que existe, este é o SUPERBASIC do Sinclair QL, uma máquina que usa o processador MC 68008 (as versões novas estão vindo com o 68020 de 32 bits!). Este BASIC possui comandos como WINDOW, REPEAT e WHILE. É uma mistura de BASIC, Pascal e Prolog, pois possui comandos das três linguagens.

O BASIC é uma linguagem muito útil para testarmos algoritmos antes de fazermos um programa diretamente em Forth, por exemplo.

Para inteligência artificial, as melhores são o Prolog e o LISP, mas nada impede que se façam programas mais simples de IA em BASIC ou Pascal, por exemplo. O problema é que um programa enorme em Pascal pode ser substituído por um programa minúsculo em Prolog.

Por incrível que pareça, o BASIC e o Pascal são boas linguagens para bancos de dados e manipulações de strings, o que é meio complicado de ser feito em Forth.

A escolha de uma linguagem adequada para a solução de um determinado problema não é algo fácil de ser feito, pois, como já disse, os fatores são muitos.

Examine os programas que constam neste artigo e, se possível, experimente-os.

LISP:

```
( DEFUN FATORIAL (N)
   (SETQ X 1)
   (DOUNTIL (COND
             ((EQ N 0) X ))
      (SETQ X (MUL X N ))
      (SETQ N (SUB N 1 ))
   )
)
```

PASCAL:

```
PROGRAM FATORIAL;
VAR N,A:INTEGER;
      B:REAL;
BEGIN
  A:=N;
  B:=1;
  WHILE A>1 DO
   BEGIN
    B:=B*A;
    A:=A-1;
   END;
  WRITELN(B);
END.
```

LOGO:

```
 TO FATORIAL : M
   MAKE : T 1
   WHILE : M > 1
   (MAKE : T : T * : M
   )
   + 1 T
   PRINT T
 END
```

BASIC:

```
10 INPUT N
20 B=1
30 FOR A=2 TO N
40 B=B*A
50 NEXT A
60 PRINT B
```

C:

```
#include <stdio.h>
long I,J,K
main()
{
printf("Entre com o numero ");
scanf("%ld",&I);
k=1;
for(J-1;J<I;J++,K*=J);
printf("Fatorial= %ld",K);
}
```

FORTH:

```
: FATORIAL 1 + 1 SWAP 1 DO I * LOOP.
```

ALGOL:

```
BEGIN
REAL     B;
INTEGER A,N;
B:=1;
FOR A:= 2 STEP1 UNTIL N DO B:=B*A;
WRITE(B);
END.
```

FORTRAN:

```
          ACCEPT N
          B=1
          DO 10 A=2,N
10        B=B*A
          TYPE B
```

APL:

```
0: FATORIAL
1: BEGIN : 'NUMERO';
2: N →.q
3: F →!N
4: F ←
5:    BEGIN
```

MATEMÁGICA

J. L. FONSECA

Vamos, hoje, apresentar uma versão do programa LIFE em duas dimensões com uma resolução que raramente é implementada em micros e que será de 256x192 pixels.

Este jogo/passatempo foi criado por John Conway e é um passatempo fascinante, principalmente na resolução aqui apresentada.

Este jogo é um autômato finito constituído por uma malha bidimensional de células que podem existir em dois estados: "vivas" ou "mortas". Uma célula morta tem a cor do fundo e uma viva tem uma cor contrastante. De uma geração para a outra as células vão nascendo e morrendo de acordo com as regras definidas abaixo.

Cada célula tem oito vizinhos que são oito pixels que a cercam e o seu estado na próxima geração é determinado pelo seu estado atual e pelo estado dos seus vizinhos. As células que estão nas bordas da tela têm como vizinhas as células da borda oposta, pelo que a tela forma uma superfície toroidal, isto é, uma suprefície em forma de "rosquinha".

Se a célula está viva e tem dois ou três vizinhos vivos, ela sobrevive na próxima geração.

Se a célula tiver mais de três ou menos de dois vizinhos, ela morre devido à superpopulação e ao isolamento, respectivamente.

Se a célula estiver morta e tiver exatamente três vizinhos vivos, ela estará viva na próxima geração.

Estas regras, embora aparentemente simples, dão origem a padrões extremamente complexos e praticamente imprevísiveis .

O programa deste mês foi feito parte em BASIC, parte em assembler, pois, devido ao número de células e ao número de testes disso decorrente, seria extremamente lento se fosse apenas em BASIC.

A parte em BASIC serve para a inicialização e para as funções de arquivo e nos apresenta quatro opções: 1 – DISTRIBUIÇÃO ALEATÓRIA; 2 – DESENHAR A TELA; 3 – SALVAR A TELA; 4 – CARREGAR A TELA.

As diversas opções são, autoexplicativas, e são acessadas digitando o seu número. Para voltar ao menu inicial basta digitar CTRL/STOP.

Na opção 2 o cursor é movido com as setas e os pontos plotados com a barra de espaço, sendo que a tecla RETURN inicia a operação do autômato.

```
30 'J.L.FONSECA
40 KEYOFF: CLS: LOCATE 8,11: PRINT"LIFE
SENDO INSTALADO..."
50 FOR I=&H9000 TO &H920F: READ A$:A=VAL
("&H"+A$): POKE I,A: NEXT I
60 ON STOP GOSUB 80: STOP ON
70 GOSUB 520: GOTO 100
80 IF A$="2" THEN DEFUSR=&H9013: A=USR(0
)
90 SCREEN 0: RETURN 100
100 CLS: LOCATE 13,1: PRINT"---LIFE---",
"         (c) BY J.L.FONSECA",,,,,,,
110 PRINT"          1-TELA ALEATóRIA",,"
      2-DESENHA TELA",,"          3-SALV
A A TELA",,"          4-CARREGA UMA TELA"
,,,"          5-RODA PROGRAMA",,"
6-FIM",,,,,,
120 A$=INKEY$: IF A$("1" OR A$)"6" THEN
120
130 OP=VAL(A$): ON OP GOSUB 150,170,450,
470,490,510
140 SCREEN0: GOTO80
150 CLS: LOCATE 10,10: INPUT "QUANTOS PO
NTOS ";A
160 SCREEN 2: FOR I=1 TO A: PSET (RND(TI
ME)*255,RND(TIME)*191): NEXTI: DEFUSR=&H
9013: A=USR(0): DEFUSR=&H9000: A=USR(0):
 DEFUSR=&H918C: GOTO 500
170 X=128: Y=96 :SCREEN 2: COLOR 15,1: C
LS: SPRITE$(0)=S0$: SPRITE$(1)=S1$
180 PUTSPRITE0,(X-3,Y-4),15,0: PUTSPRITE
1,(X-3,Y-4),1,1
190 I=STICK(0): ON I GOSUB 250,270,300,3
20,350,370,400,420
200 I=STRIG(0): IF I()0 THEN GOSUB 220
210 GOTO 180
220 IF POINT(X,Y)=1 THEN PSET(X,Y),15 EL
SE PSET(X,Y),1
230 FOR I=1 TO 50: NEXT I
240 RETURN
250 Y=Y-1: IF Y(0 THEN Y=0
260 RETURN
270 Y=Y-1: IF Y(0 THEN Y=0
280 X=X+1: IF X)255 THEN X=255
290 RETURN
300 X=X+1: IF X)255 THEN X=255
310 RETURN
320 Y=Y+1: IF Y)191 THEN Y=191
330 X=X+1: IF X)255 THEN X=255
340 RETURN
350 Y=Y+1: IF Y)191 THEN Y=191
360 RETURN
370 Y=Y+1: IF Y)191 THEN Y=191
380 X=X-1: IF X(0 THEN X=0
390 RETURN
400 X=X-1: IF X(0 THEN X=0
410 RETURN
420 Y=Y-1: IF Y(0 THEN Y=0
430 X=X-1: IF X(0 THEN X=0
440 RETURN
450 CLS:LOCATE 10,10: PRINT"NOME DO ARQU
IVO ?": LOCATE 10,12: LINEINPUT A$
460 BSAVE A$,&H920F,&HAA0F: GOTO 80
470 CLS:LOCATE 10,10: PRINT"NOME DO ARQU
IVO ?": LOCATE 10,12: LINEINPUT A$
480 BLOAD A$: GOTO 80
490 SCREEN 2: DEFUSR=&H9000: A=USR(0): D
EFUSR=&H903D: A=USR(0): DEFUSR=&H918C
500 A=USR(0): GOTO 500
510 END
520 S0$=CHR$(16)+CHR$(16)+CHR$(16)+CHR$(
238)+CHR$(16)+CHR$(16)+CHR$(16)+CHR$(0):
 S1$=CHR$(40)+CHR$(40)+CHR$(238)+CHR$(0)
+CHR$(238)+CHR$(40)+CHR$(40)+CHR$(0)
530 RETURN
540  DATA 21,00,20,3E,F1,0E,18,06
550  DATA 00,CD,CD,07,23,10,FA,0D
560  DATA 20,F5,C9,21,00,00,11,08
570  DATA 08,DD,21,0F,92,0E,C0,06
580  DATA 20,E5,CD,D7,07,DD,77,00
590  DATA DD,23,7B,85,6F,10,F3,E1
600  DATA 2C,15,20,05,24,2E,00,16
610  DATA 08,0D,20,E3,C9,21,00,00
620  DATA 11,08,08,DD,21,0F,92,0E
630  DATA C0,06,20,E5,DD,7E,00,CD
640  DATA CD,07,DD,23,7B,85,6F,10
650  DATA F3,E1,2C,15,20,05,24,16
660  DATA 08,2E,00,0D,20,E3,C9,7B
670  DATA FE,FF,20,02,3E,BF,FE,C0
680  DATA 20,02,3E,00,6F,26,00,29
690  DATA 6F,26,00,29,29,29,29,29
700  DATA 7A,CB,3F,CB,3F,CB,3F,85
710  DATA 6F,30,01,24,7A,E6,07,11
720  DATA 07,92,83,5F,1A,11,0F,92
730  DATA 19,C9,ED,5B,05,92,0E,00
740  DATA 15,2A,03,92,CD,80,90,47
750  DATA A6,28,01,0C,11,20,00,19
760  DATA 78,A6,28,01,0C,19,78,A6
770  DATA 28,01,0C,ED,5B,05,92,2A
780  DATA 03,92,CD,80,90,47,A6,28
790  DATA 01,0C,11,40,00,19,78,A6
800  DATA 28,01,0C,ED,5B,05,92,14
810  DATA 2A,03,92,CD,80,90,47,A6
820  DATA 28,01,0C,11,20,00,19,78
830  DATA A6,28,01,0C,19,78,A6,28
840  DATA 01,0C,C9,ED,5B,05,92,15
850  DATA 1D,CD,67,90,A6,28,01,0C
860  DATA ED,5B,05,92,1D,CD,67,90
870  DATA A6,28,01,0C,ED,5B,05,92
880  DATA 14,1D,CD,67,90,A6,28,01
890  DATA 0C,ED,5B,05,92,15,CD,67
900  DATA 90,A6,28,01,0C,ED,5B,05
910  DATA 92,14,CD,67,90,A6,28,01
920  DATA 0C,ED,5B,05,92,1C,15,CD
930  DATA 67,90,A6,28,01,0C,ED,5B
940  DATA 05,92,1C,CD,67,90,A6,28
950  DATA 01,0C,ED,5B,05,92,14,1C
960  DATA CD,67,90,A6,28,01,0C,C9
970  DATA ED,5B,05,92,CD,67,90,11
980  DATA 00,18,19,B6,77,C9,06,00
990  DATA C5,CD,F3,90,79,FE,03,CC
1000  DATA 5B,91,FE,02,20,08,ED,5B
1010  DATA 05,92,CD,67,90,A6,C4,5B
1020  DATA 91,3A,06,92,3C,32,06,92
1030  DATA C1,10,DD,C9,11,00,00,ED
1040  DATA 53,03,92,ED,53,05,92,11
1050  DATA 0F,AA,0E,18,3E,00,06,00
1060  DATA 12,13,10,FC,0D,20,F7,CD
1070  DATA 66,91,3A,05,92,3C,32,05
1080  DATA 92,0E,BE,06,00,C5,CD,9A
1090  DATA 90,79,FE,03,CC,5B,91,FE
1100  DATA 02,20,08,ED,5B,05,92,CD
1110  DATA 67,90,A6,C4,5B,91,3A,06
1120  DATA 92,3C,32,06,92,C1,10,DD
1130  DATA 2A,03,92,11,20,00,19,22
1140  DATA 03,92,3A,05,92,3C,32,05
1150  DATA 92,0D,20,C7,3E,BF,32,05
1160  DATA 92,CD,66,91,11,0F,92,21
1170  DATA 0F,AA,01,00,18,ED,B0,CD
1180  DATA 3D,90,C9,00,00,00,00,80
1190  DATA 40,20,10,08,04,02,01,00
```

Cartas

Lendo o artigo entitulado "Slots e expansões", de autoria do Sr. André L. de Freitas, despertou-me a curiosidade de escrever e parabenizá-los por esta edição maravilhosa e por este artigo, no qual eu desejo aprofundar-me com a ajuda do colaborador.

Gostaria que o André me ajudasse a resolver um velho problema meu, e acredito que de muita gente.

Há algum tempo adquiri um software muito bom chamado "Graphos III". Ao carregá-lo, editei uma tela muito bonita, a qual pretendia incluí-la em um programa meu em Basic, terminando de editá-la pelo "Graphos III".

Tratei de salvá-la em uma outra fita cassete virgem, sendo que no menu "Arquivo Tela" há opções para salvar com as seguintes opções:

- Display
- Lay Out
- Compac

Ao arquivá-la nos três modos acima, não consegui chamar a tela de volta por um programa em Basic, ou melhor, não consegui criar uma rotina em Basic que pudesse aplicar esta tela em um dos meus programas.

Por isso, gostaria que me ajudasse a resolver este problema, o qual já me tirou algumas horas de sono.

A tela só é recuperada pelo próprio "Graphos III".

Utilizando um capturador de "Header", verifiquei que todas as telas salvadas possuem como endereço inicial &h9200. Verifiquei, também, que o endereço inicial do "Graphos III" é &h8400 e termina no endereço &hC500, tendo como endereço de entrada &h922e.

Portanto, pelo seu artigo, estas telas devem estar armazenadas em outra página da memória. Não é isso?

Só que sou "marinheiro de primeira viagem" e não estou entendendo mais nada. Me ajude. Aguardo qualquer resposta, anciosamente.

Meu Micro é um Expert 1.1 e utilizo somente data corder.

José Laudomiro G. Moura
End.: Q-G C-5
Bairro Parque Piauí
Teresina – PI
Cep: 64000

Gostaria de uma informação sobre as telas editadas no programa GRAPHOS III. Quando a tela é arquivada no disquete, é selecionado um nome e este nome ganha, automaticamente, uma extenção .SCR. No meu caso, a título de exemplo, digamos que eu tenha uma tela com o nome no arquivo de < TELA.SCR >. Normalmente , quando quero visualizar esta tela, digito o seguinte programa:

```
10 SCREEN 2
20 BLOAD"TELA.SCR",R
30 GOTO 30
```

Obviamente, a linha 30 é para congelar a tela no vídeo. Bem, até aí, tudo certo, mas o que eu quero saber é como acessar ou chamar esta mesma tela, sabendo que ela está na memória do micro? Ou se eu digitar BLOAD"TELA.SCR" sem o, (R) e, logo após, RETURN, aparece o OK. Como acessar sem acionar o drive, ou seja, somente através de comando?

Esperando que possa ceder-me esta "dica", fico aguardando a resposta.

Antônio Carlos Tavares de Almeida
Rua Candida Tavares Silveira, 131/102
Nilópolis – RJ
26510

Este mês responderei a duas cartas enviadas pelos leitores José Moura, de Terezina, Piauí e Antônio Carlos Almeida, de Nilópolis, Rio de Janeiro, onde pediam a solução para o carregamento de telas editadas pelo programa GRAPHOS III.

A dúvida do leitor José Moura era a seguinte: Como carregar pelo BASIC, as teclas editadas, salvas em fita cassete?

Antes de carregar a tela, certifique-se de estar na SCREEN 2. Caso contrário, a tela será carregada, mas na hora de ser tranportada para a memória de vídeo (isto é feito automaticamente pelo bloco salvo na fita, no modo DISPLAY) causará uma grande confusão ao processador de vídeo, não aparecendo nada do esperado na tela.

Um exemplo dos programas para você testar o carregamento é o seguinte:

```
10 SCREEN 2
20 BLOAD "CAS:",R
30 GOTO 30
```

Isto carregará a tela, sendo que a linha 30 é uma "armadilha" para o micro não voltar a SCREEN 0.

O leitor gostaria de saber, também, se a tela estaria em outro SLOT do micro. Isto não acontece, a tela é carregada dentro da área do BASIC. Na própria RAM no SLOT usado pelo BASIC.

Já o leitor Antônio Carlos Almeida gostaria de saber como carregar a tela da fita sem a mesma ser inicializada automaticamente. Isto é possível, mas devido ao fato dela ser carregada dentro da área do BASIC, se o programa carregador em BASIC for muito grande, haverá um conflito entre os dois. Se o programa que usará a tela for em assembly, tome cuidado para não usar os mesmos endereços para ele e a tela.

Para carregar use o BLOD, sem a opção "R", e, para gerar a tela a qualquer momento, use DEFUSR=&H9200 e X=USR(0).

Lembre-se de estar na SCREEN 2.

os endereços de uma tela salva no GRAPHOS III no modo DISPLAY são:

Início:	&h9200
Final:	&HC280
Entrada:	&H9200

Espero que as dúvidas estejam satisfeitas.

André L. F. de Freitas

Comandos do MSX-DOS

ANDRÉ L. F. DE FREITAS

Com o aparecimento do Disk Driver para a linha MSX no Brasil, veio com o mesmo o sistema controlador deste periférico, o chamdo DOS (Disk Operation System), sendo o MSX DOS um software controlador desenvolvido pela empresa americana Microsoft e rapidamente adaptado por outras empresas que comercializavam drives para a linha MSX.

Aliás, eu gostaria de saber se alguma empresa nacional possui o direito de venda deste software, dado pela Microsoft, com contratos e etc., ou somente usam o artifício de registrar e comercializar no Brasil produtos a muito custo desenvolvidos no exterior, tornando-se piratas de baixíssimo nível, dizendo muitas vezes que o DOS comercializado por elas é de sua própria autoria.

Além de tudo isso, ainda existem empresas, como a Microsol, que conseguiram uma façanha inédita no desenvolvimento de sistemas operacionais. O DOS da Microsol possui um arquivo único de sistema, o SOLXDOS.SIS, que contém, simultaneamente, todas as rotinas do DOS e o interpretador de comandos e, além disso, comandos em português, tornando-se a primeira empresa a comercializar um DOS bilingue (essa é terrível !).

Se você quiser desenvolver o seu próprio interpretador de comandos, que é uma das opções que se tem se o sistema possuir o arquivo COMMAND.COM independente, não poderá fazê-lo usando o sistema da Microsol. Seria isso uma forma de evitar pirataria de programas ou reprimir o usuário capaz de desenvolver algo melhor?

Mas, esquecendo as confusões surgidas desde que começou o desenvolvimento de software nacional, vamos ao que interessa a respeito do DOS.

O sistema operacional MSX DOS foi desenvolvido para gerenciar todas as operações de disco realizadas pelo MSX. Um não pode existir sem o outro, a menos que o leitor seja um excelente programador para criar seu próprio software de operação de disco, que, na verdade, também seria um DOS.

O MSX DOS é um sistema operacional baseado no CP/M versão 2.2, apesar de se parecer muito com o MS DOS, da linha IBM PC. Possui comandos do MS DOS, mas, no que diz respeito às rotinas internas em assembly, as chamadas das mesmas, valores passados e recebidos por registradores, assemelha-se ao CP/M. O tratamento destas rotinas ficará a cargo de outra parte desta série de artigos.

Começaremos então, com a descrição dos sistema. Quando ligamos nosso MSX com um disk drive conectado, percebemos a inicialização normal do micro e, logo após, surge uma mensagem do fabricante da interface. Se não houver disco com sistema operacional no drive, o micro passa direto ao Disk BASIC, ou, então, solicita um diskette contendo sistema operacional (isto vai depender do fabricante de sua interface).

Supondo que há um disco com sistema no drive, o mesmo continuará a ser lido e logo surge uma mensagem no vídeo dizendo a versão do DOS, o criador do software, etc. A seguir, surgirá um sinal: A>, indicando que o drive corrente em uso é o drive A.

Este sinal "A>" é chamado PROMPT e significa que o DOS aguarda a digitação de um comando. Não traduzirei os termos usados neste artigo para não criar confusões.

O MSX DOS possui um grande número de comandos, os quais relacionarei abaixo, apresentando sua sintaxe e utilização, pois existem algumas formas que talvez não sejam do conhecimento de todos.

Antes de começar, porém, mostrarei algumas particularidades que não são comandos, mas devem ser obedecidas.

PARTICULARIDADES DO DOS

- Os drives são designados por letras (A-F) não havendo diferenciação entre maiúsculas ou minúsculas, sendo permitidos somente aqueles presentes ao sistema. O primeiro será o A, o segundo o B e assim por diante.
- Para a mudança do drive em uso, basta diigitar a letra correspondente seguida pelo caracter "dois pontos" e teclar [RETURN] ao final. Como exemplo, para passar a usar o drive B, se existir, basta teclar: B: [RETURN]
- Os nomes de arquivos possuem um máximo de 11 caracteres separados em 2 grupos por um "ponto", sendo o primeiro grupo de até 8 caracteres, significando o nome do arquivo, e o segundo grupo opcional, contendo no máximo 3 caracteres, chamado de extensão do arquivo, servindo para identificar o tipo de arquivo usado. Ex.: NOMEARQ1.EXT
- No nome do arquivo podem ser utilizados quaisquer caracteres de A até Z, dígitos de 0 a 9, e alguns símbolos do teclado do micro, devendo começar com uma letra, não sendo permitido o caracter "espaço", havendo uma mensagem de erro quando o caracter não for válido. Também aqui não há diferenciação entre maiúsculas ou minúsculas, sendo que o micro converte tudo automaticamente para maiúsculas.
- Os nomes podem ser os nomes de comandos internos do DOS. A lista de comandos internos é a seguinte:

BASIC	VERIFY
DIR	DATE
PAUSE	FORMAT
TYPE	RENAME
COPY	DEL
ERASE	MODE
REM	TIME

- Também não podem ser usados como nomes de arquivos os nomes utilizados para designar dispositivos do micro:

AUX – significa dispositivo auxiliar (interface externa como impressora, drive, etc.)

CON – significa console (teclado ou vídeo)

NUL – dispositivo nulo. Dispositivo sem efeito no micro.

LST ou

PRN – impressora.

Estas palavrras são tratadas como arquivos especiais para redirecionamento de saídas ou entradas.

- Caracteres especiais: * (assterisco) e ? (ponto de interrrogação).

O primeiro é usado para generalizar um grupo de caracteres quaisquer nos nomês de arquivo. Por exemplo:

Nome*.* pode referenciar qualquer arquivo começando com MOME, possuindo no máximo mais 4 caracteres e qualquer extensão.

NOMEARQ.* pode referenciar qualquer arquivo de nome MOMEARQ possuindo qualquer extensão.

O caracter ? é usado da mesma forma que o primeiro, sendo que só substitui 1 caracter do nome do arquivo. Ex:

NOMEARQ?.??T referencia arquivos com quaisquer caracteres nos lugares onde se encontram interrogações.

Procure sempre usar nomes para seus arquivos que indiquem exatamente o que eles representam. Por exemplo, GRAFICOS.BAS seria um bom nome para um programa em BASIC que gerasse gráficos. As extensões mais usadas comercialmente seriam as seguintes:

BAS – programas em BASIC
BIN – programas em linguagem de máquina (acesso via BASIC)
FOR – programas em FORTRAN
ASM – assembly
PAS – pascal
C – linguagem C
LIB – arquivos de biblioteca de certas linguagens
DOC – textos em modo documento
TXT – textos
COM – programas diretamente executáveis

Procure usar extensões que não passem informações sobre o arquivo.

COMANDO DO DOS

Os comandos do DOS são formados, geralmente, por uma palavra, o comando propriamente dito seguido de uma ou mais palavras, chamadas de parâmetros, com um [RETURN] teclado ao final. Para serem interrompidos, basta pressionar, simultaneamente, as teclas [CONTROL] e [C]. Isto causará o encerramento imediato, se possível, do comando que está sendo executado. Outras teclas desempenham funções especiais no DOS, mas isso ficará para a segunda parte do artigo, no próximo número.

Utilizarei a seguinte convenção para a descrição abaixo:

Toda vez que uma palavra estiver entre os sinais <e>, a mesma é opcional.

Quando aparecer X: ou Y: isto significará qualquer drive existente no sistema.

Use espaços somente para separar parâmetros uns dos outros.

COMANDOS

BASIC
Permite a entrada no DISK BASIC MSX

Sintaxe:
BASIC <X> NOMEARQ.BAS>
Caso seja fornecido o nome do arquivo, o mesmo será executado automaticamente na entrada do BASIC.
Para voltar ao sistema operacional, basta digitar CALL SYSTEM, dentro do BASIC.

COPY:
Faz cópias de arquivos entre discos. Permite juntar arquivos em um só enquanto copia.

Sintaxe:
COPY <X:><ARQUIVO1.EXT><X:><ARQUIVO2.EXT>

Exemplo:
COPY A:COMMAND.COM B:COMANDOS.COM
Copia o arquivo COMMAND.COM do drive A para o B com o novo nome COMANDOS.COM
COPY TESTE.BAS B:
Copia o arquivo TESTE.BAS no drive em uso para o drive B com o mesmo nome.

OBS1:
São válidos os caracteres especiais * e ?
COPY TESTE.* B:
copiará todos os arquivos TESTE com qualquer extensão para o drive B.

OBS2:Se for usado um nome de dispositivo no lugar do segundo nome de arquivo, a saída será por aquele dispositivo.
COPY TESTE.BAS PRN
listará o arquivo na impressora
COPY TESTE.BAS CON
listará o arquivo no vídeo.

OBS3: Para concatenar (juntar) arquivos durante a cópia, basta referenciá-los como se fossem somados no nome do primeiro arquivo.
COPY A.DOC+B.DOC+C.DOC FINAL.DOC
copiará os arquivos A.DOC, B.DOC e C.DOC um após o outro para dentro do arquivo FINAL.DOC

DATE
Mostra e permite alterar a data do sistema.

Sintaxe:
DATE <DD/MM/AA>

Exemplo:
DATE 03/06/88
Altera a data para 03/06/88.
DATE
Mostra a data atual e pede uma nova data. Se for teclado [RETURN] somente, a data é alterada.

DEL (ou ERASE)
Permite apagar do disco arquivos que não utilizaremos mais. O comando ERASE é idêntico.

Sintaxe:
DEL <X:>NOMEARQ.EXT
ERASE <X:>NOMEARQ.EXT
apaga o arquivo especificado do disco contido no drive especificado. Não fornecido o drive, o mesmo é assumido como o corrente em uso.

OBS: São válidos os caracteres * e ?
DEL *.*
apagará TODOS os arquivos no disco. O sistema perguntará se realmente você quer fazer isso, pois depois de apagados os arquivos não são mais recuperáveis a não ser por alguns programas "milagrosos".
DEL *.BAS
apagará somente os arquivos com extensão BAS no disco. A pergunta para confirmar a deleção não é mais feita, portanto, cuidado.

DIR
Permite observarmos os arquivos existentes no disco. O comando mostrará nomes, extensões, data em que o arquivo foi criado, hora (em alguns sistemas), tamanho em bytes, e, ao final, o número de arquivos e o espaço livre no disco.

Sintaxe:
DIR <X:><NOMEARQ.EXT></W></P>

Exemplo:
DIR
Mostrará todos os arquivos do disco.
DIR *.COM
Mostrará todos os arquivos do disco com extensão COM
DIR/W
Apresentará somente os nomes dos arquivos em duas colunas no vídeo.
DIR/P
Se a listagem dos arquivos ultrapassar o tamanho da tela, esta opção fará uma pausa aguardando uma tecla para continuar sempre que a tela for preenchida.

FORMAT
Permite formatar, ou seja, preparar um disco virgem para receber informações.

Sintaxe:
FORMAT

OBS1: A seguir virão perguntas sobre o tipo e capacidade do disk driver que você possui, devendo ser cuidadosamente respondido.

OBS2: Cuidado para não formatar discos que contenham programas, pois todo o conteúdo dos mesmos estará perdido para sempre.

MODE
Seleciona modo de tela e colunas no vídeo.

Sintaxe:
MODE <largura>

Exemplo:
MODE 40

OBS: Caso <largura> seja maior do que 32, o modo do vídeo será equivalente ao modo 0 do BASIC.
Caso seja menor, o modo do vídeo será equivalente ao modo 1.
<largura> deve ser um número entre 1 e 40.

TIME

Permite atualizar a hora do sistema. Só produz efeito se o seu micro possuii um relógio interno.
Sintaxe:
TIME <HH:MM:SS>
Exemplo:
TIME 13:30:00

Atualiza a hora para 13:30 (os segundos podem ser omitidos, sendo então zerados automaticamente).

TIME

Mostra a hora corrente e pede nova hora. Idem ao comando DATE.

TYPE

Mostra o conteúdo de um arquivo texto.
Sintaxe:
TYPE NOMEARQ.EXT
Exemplo:
TYPE ARTIGO.CPU
Mostra o conteúdo do artigo ARTIGO.CPU no vídeo.

VERIFY

Permite selecionar modo de verificação automática em um arquivo que acabou de ser escrito no disco para testar a ocorrência de erros.
Sintaxe:
VERIFY <ON/OFF>
VERIFY ON liga o modo de verificação.
VERIFY OFF desliga.

OBS: Com este comando acionado, a escrita em disco se tornará mais lenta, porém mais confiável, acusando erro sempre que houver.

Os comandos PAUSE e REM serão tratados na segunda parte deste artigo, no próximo número da revista, pois requerem um conhecimento de arquivos do tipo BATCH, dos quais irei tratar com mais detalhes. Veremos, também, controles especiais do DOS, como teclas com funções importantes e mais algumas curiosidades, antes de começarmos a tratar das funções do DOS a nível de linguagem de máquina.

Esta primeira parte de nossa aventura pelo sistema operacional pode ser elementar para usuários experientes, mas não poderia faltar, pois nosso objetivo é descrever o DOS em todos os sentidos, dedicando o conhecimento a todos os usuários, quer sejam "pequenas crianças conhecendo o mundo da informática" quer sejam "feras da programação".

MSX DOS e MS DOS são marcas registradas de Microsoft Corp.
CP/M é marca registrada da Digital Research Corp.
IBM PC é marca registrada da IBM Corp.

Nemesis

O jogo NEMESIS, da Konami, é, sem dúvida alguma, um dos melhores jogos espaciais existentes para o MSX 1.0, rodando somente em disco de face dupla ou em cartucho

Para ajudá-lo a desvendar este maravilhoso jogo, damos, abaixo, algumas dicas extraídas do manual que acompanha o cartucho.

O JOGO

O planeta Nemesis é um mundo pacífico semelhante à Terra, mas, no momento, está sendo atacado pelos seus inimigos de sempre, os Bacterions. O povo de Nemesis está ameaçado de ser completamente destruído pelo povo de Bacterion.

Seu objetivo, portanto, é ajudar o povo de Nemesis a enfrentar esta terrível ameaça. Para tal, uma nave foi desenvolvida, a Warp Rattler. Conquistando a super-fortaleza dos Bacterion, a Xacrous, você terá salvo o povo de Nemesis.

COMO JOGAR

O jogo pode ser jogado por um ou dois jogadores. No caso de dois jogadores, cada um joga alternadamente.

Os movimentos da nave podem ser controlados pelo joystick ou utilizando-se as teclas de controle do cursor e a barra de espaços.

Os caças inimigos irão atacar continuamente sua nave. Você tem que eliminar o maior número possível de caças inimigos para poder passar para o estágio seguinte.

Sua nave é rápida mas o seu armamento não é suficiente para enfrentar com sucesso os ataques inimigos. Você tem que receber constantes aumentos de força através das cápsulas de força do inimigo.

Ao iniciar o jogo, você possui 3 Warp Rattlers e, a cada 100.000 pontos conseguidos, você receberá outra nave. O número máximo de naves que poderão ser conseguidas é de 99.

Uma nave será perdida sempre que for atingida por fogo inimigo, tocar em naves inimigas ou em qualquer obstáculo do solo.

Para parar momentaneamente o jogo, pressione F1, pressionando-a novamente para retornar.

PONTUAÇÃO

Destruição de uma nave pequena inimiga: 100 pontos

Destruição de uma média inimiga: 1.000 pontos

Destruição da parte central de um monstro: 10.000 pontos

Destruição de inimigos diversos até: 5.000 pontos

AUMENTO DE FORÇA

Cada vez que você pega uma cápsula de força do inimigo, alguns indicadores de seleção de aumento de força, na parte inferior da tela, vão se iluminar. Quando desejar selecionar o campo iluminado, pressione as teclas "m" ou "n" no teclado, ou botão B do joystick.

Se você apanhar uma cápsula de força azul, todos os inimigos presentes na tela serão destruídos imediatamente.

AJUDA ADICIONAL

Como em todos os jogos da Konami, o Nemesis também possui alguns macetes que poderão ajudá-lo a conquistar o seu objetivo mais facilmente.

Para ter acesso às senhas, pare o jogo, pressionando a tecla F1.

Digite:
hyper + RETURN
Para voltar a jogar, pressione novamente F1.

Além do Hyper, temos, ainda, FIM, LASER, MISSILE, SHIELD, OPTION, DOUBLE e LUCIENE.

O nome da senha pode variar de acordo com a versão do jogo que você possui, pois algumas softhouses efetuaram alterações. Caso o seu jogo não aceite estas senhas, utilize um Zapper para verificar os nomes que foram atribuídos às senhas.

Elas poderão ser encontradas no primeiro bloco do jogo, sem contar a capa de apresentação da softhouse, geralmente o bloco de nome NEME1, nos setores 35 e 36.

The castle

MSX INFORMÁTICA

The Castle é um dos mais emocionantes jogos de labirinto para o MSX.

Sua estória se passa num castelo medieval, onde você deve ajudar o príncipe Rafael a salvar sua amada, a princesa Margarida, que foi raptada e está, aprisionada na torre do castelo do cruel Mephisto. Para tanto, guiará o valente príncipe através das cem salas do castelo até a torre.

Com um visual agradável e um fundo musical que a acompanha a velocidade em que estiver jogando, é um jogo que desafia e estimula sua imaginação, o que fará você continuar jogando na tentativa de vencer.

Desde o seu lançamento, tem sido muito procurado pelos usuários do MSX, figurando sempre na lista dos melhores jogos. Tanto adultos como crianças não poupam esforços para alcançar o fim do jogo.

Por isso, achamos que valia a pena descrevê-lo detalhadamente, além de dar algumas dicas que ajudam a vencer.

1 – Para jogar

Você pode utilizar joystick ou o teclado (setas cursoras e barra de espaço).

2 – Seus inimigos

Não deixe nenhum inimigo tocar em você; caso contrário, você morrerá.

Conheça, agora, os seus inimigos:

– Os guardas do castelo: usam armadura

– Os escravos: vestem um colete

– As bruxas: têm bengala e chapéu

O raposão: tem uma longa barba e uma roupa vermelha

– As bolas de fogo

– As aranhas (quando uma se transformar num pequeno triângulo, passe por cima dela. Não esqueça de teclar espaço ou o botão de tiro).

Você poderá matá-los jogando objetos (tijolos, jarros, cofres e bolos) em cima deles ou, então, atropelando-os com tais objetos.

Eles também morrem esmagados em elevadores, já que não passam de uma sala para outra.

3 – Fique atento

● Tente pegar o maior número de chaves e com elas abrir o menor número de portas (você pode perder o jogo se cair numa sala e não tiver as chaves para sair dela).

● Antes de deixar um objeto (ex. tijolo) ser esmigalhado pelo elevador, verifique se não vai precisar dele para pegar algum outro objeto ou mesmo sair da sala, pois, uma vez esmigalhados, não reaparecem mais. Caso você realmente precise dele e não tenha como pegá-lo, tecle F1 antes que o elevador destrua o objeto (você perderá uma vida, mas o objeto voltará a sua posição original).

● Enquanto o campo magnético estiver em movimento, você poderá caminhar por cima dele. Quando ele pará, você pode atravessá-lo de um lado para o outro. Tome cuidado se, no momento em que estiver atravessando, o campo voltar a se movimentar; você morre.

● A cada 10000 pontos você ganha uma vida.

4 – Objetos úteis

O MAPA DO CASTELO: O mapa localiza-se na sala inicial. Quando você pegá-lo, aparecerá no alto da tela um mapa que indicará a sala onde você está e marcará as salas que já foram percorridas.

POÇÃO MÁGICA: Ao beber a poção da garrafa verde, Rafael ganha mais uma vida.

CHAVES: amarela - 10 pontos; azul clara - 20 pontos; verde - 50 pontos; lilás - 100 pontos; azul real - 400 pontos.

FADA: Ao libertar a fada, você ganhará uma chave vermelha que serve para abrir a porta da sala onde a princesa é prisioneira.

JÓIAS: barras de ouro - 800 pontos; anel - 400 pontos; cruz - 200 pontos.

OXIGÊNIO; Use-o para atravessar as regiões alagadas (Cuidado! A duração do ar é por tempo limitado!).

RAIO COLORIDO: Ao passar por ele, você fica invisivel por algum tempo (nada nem ninguém poderá matar você).

CEREJAS: Se você ficar tentado a pegá-las, vá em frente, mas fique atento para não cair vários andares abaiixo.

BARRAS DE FERRO QUE VOAM: Servem para transportar você de um lado para outro das salas; porém, tome cuidado para não ser atravessado por uma delas.

– Os segredos do teclado

CONTROL – Mantendo esta tecla pressionada você poderá jogar na velocidade rápida.

CONTROL+LGRA – Mantendo estas duas teclas pressionadas simultaneamente, você pode jogar na velocidade ultra-rápida. (É muito difícil jogar nesta velocidade, por isso procure usá-la para subir/descer os elevadores mais rápido).

TECLAS DE FUNÇÃO

F1 – Ao teclar F1, você comete suícidio e a sala volta ao estado original, ou seja, você perde as chaves e objetos que eventualmente tenha pego.

F2 – Termina o jogo.

F4 – Grava o jogo (em fita cassete) no estágio em que você estiver.

CONTROL+F3 – Ao pressionar estas duas teclas, aparecerá na tela a opção pra você chamar o jogo gravado em cassete através da F4.

6 – As salas mais difíceis

As ilutrações mostram como vencer algumas das salas mais difíceis do jogo.

Numere o mapa da esquerda para a direita e de baixo para cima de 1 até 10 para poder localizar-se.

As salas correspondentes às gravuras são: sala 6,4 (coluna, linha); sala 8,5; sala 4,7; sala 2,10;

As salas que exigem mais rapidez que astúcia são: sala 10,6; sala 3,6.

Se ainda assim você tiver dificuldade para vencer todas as etapas do jogo, siga as instruções abaixo para ficar com 255 vidas.

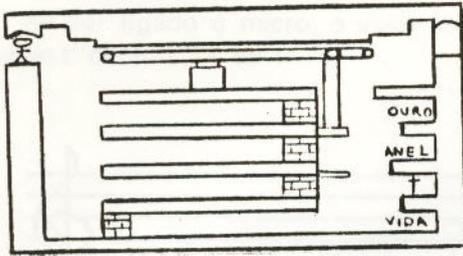
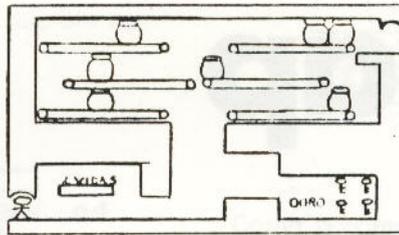
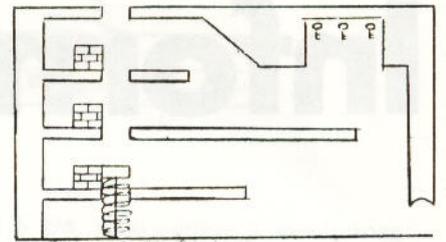
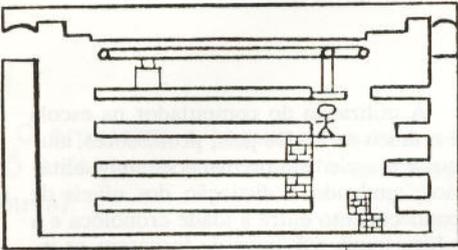
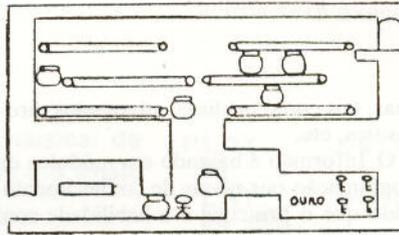
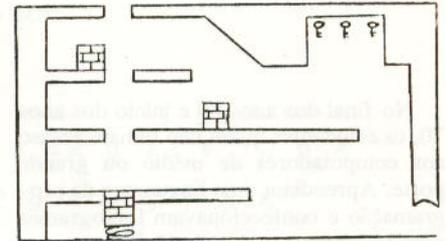
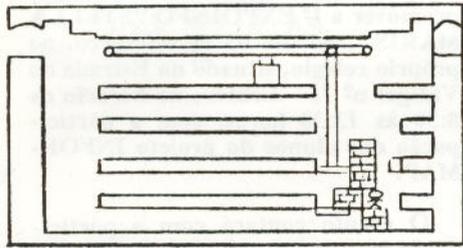
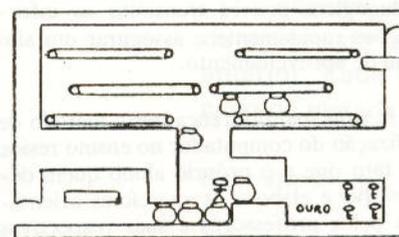
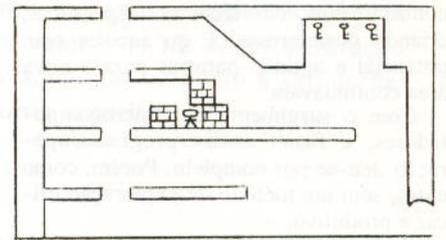
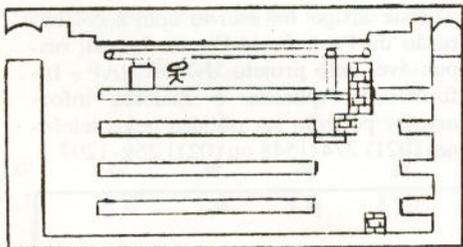
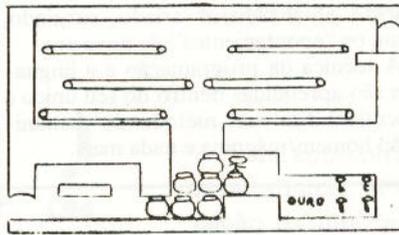
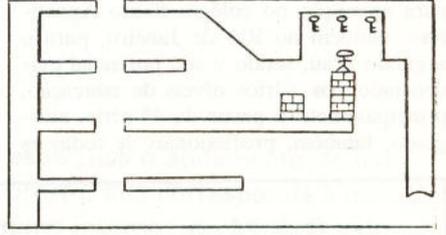
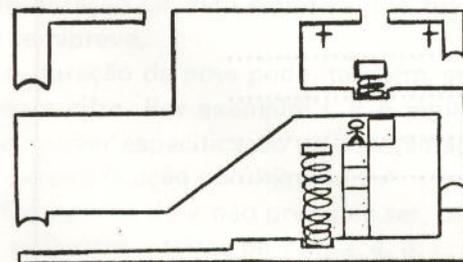
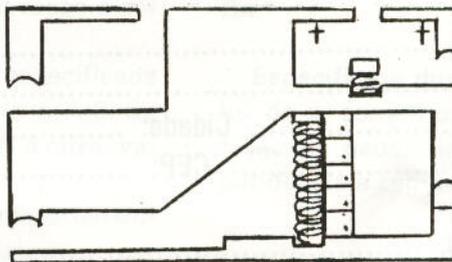
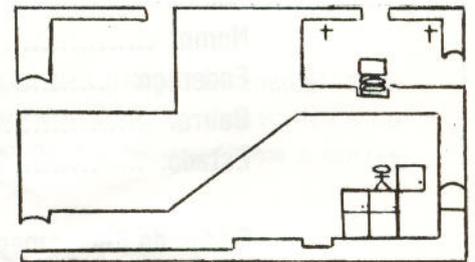
Vá para o BASIC.

O jogo The Castle (em disco) vem dividido em dois blocos. No meu caso, o nome destes blocos é: CASTLE1.BIN E CASTLE2.BIN. Verifique o nome dos blocos de seu jogo e digite o seguinte programa colocando os nomes correspondentes ao seu programa, é claro.

```
10 BLOAD"CASTLE1.BIN": POKE &H9D53,240:DEFUSR= &HD000: PRINT USR(0): BLOAD "CASTLE2.BIN",R
```

Após ter digitado o programa, salve-o digitando: SAVE"CASTLINF.BAS"

De agora em diante, para jogar, basta digitar: RUN"CASTLINF.BAS"

Caso você possua fita cassete, substitua o nome do programa por "CAS:"

SALA 2,10 A — SALA 4,7 A — SALA 6,4 A

SALA 2,10 B — SALA 4,7 B — SALA 6,4 B

SALA 2,10 C — SALA 4,7 C — SALA 6,4 C

SALA 2,10 E — SALA 4,7 D — SALA 6,4 D

SALA 8,5 A — SALA 8,5 B — SALA 8,5 C

Informap

PROF. FARID

No final dos anos 60 e início dos anos 70, os estudantes quase não tinham acesso aos computadores de médio ou grande porte. Aprendiam uma linguagem de programação e confeccionavam fluxogramas para exercitar o racíocinio lógico.

As dificuldades eram enormes. Para aprender a programar era necessário ter o computador. Para poder programar era necessário saber operá-lo, o que exigia o conhecimento de técnicas específicas, criando desinteresse. E so aqueles com potencial e aptidão naturais para a nova área continuavam.

Com o surgimento dos microcomputadores, a fusão análise/programa/operação deu-se por completo. Porém, como antes, sem um método de exploração eficaz e produtivo.

INFORMAP é um programa de treinamento objetivo eficaz, estando sendo aplicado no Rio de Janeiro no Instituto Educacional Stella Maris (1º grau), em caráter optativo, e sendo desenvolvido para aplicação no colégio Santo Agostinho, também no Rio de Janeiro, para o segundo grau, sendo o seu potencial evidenciado nos vários níveis de educação, principalmente a partir da 4ª série, atingindo, também, profissionais de todas as áreas, tais como médicas, advogados, professores, etc.

O Informap é baseado em módulos de programação por níveis de conhecimento, sendo que o principal é a habilidade com que são escritos os programas e o seu nível de abrangência, pois um programa de ensino mal elaborado pode ser cansativo e irritante, enquanto que um programa preparado com conhecimentos psicológicos e pedagógicos poderá transmitir as informações rapidamente e assegurar um alto grau de aproveitamento.

A principal diferença neste método de utilização do computador no ensino reside no fato que é o próprio aluno quem desenvolve e elabora os exercícios orientados pelos professores e suas respectivas disciplinas, ou seja, o computador é utilizado como um segmento da sala de aula convencional.

Através dos alunos e dos professores, um poderoso banco de dados, permanentemente atualizado, é criado, surgindo, assim, os "apontamentos" da nova era.

A técnica da programação e a linguagem são aprendidas dentro do seu único e indicútivel fim: um método de comunicação homem/máquina e nada mais.

A utilização do computador na escola é o único meio dos pais, professores, alunos e a sociedade premanecerem igualitários, anulando a distinção dos níveis de conhecimento entre a idade cronolóca e a educacional, sob pena de tornarem-se irremediavelmente obsoletos.

No mês de setembro próximo, o Instituto Educacional Stella Maris irá promover a 1ª EXPOINFO "STELLA MARIS", no dia 03 de setembro, no próprio colégio, situado na Estrada do Vidigal nº 75 – Leblon, no horário de 8:30 às 13:30 horas, com a participação dos alunos do projeto INFORMAP.

O evento contará com a participação da Gradiente, através da doação de vários cartuchos de jogos e da revista CPU, que irá promover os melhores programas apresentados.

Este artigo foi escrito com a colaboração da Prof. Farid Facure Lahud, responsável pelo projeto INFORMAP – Informática Aplicada e maiores informações poderão ser obtidas pelos telefones (021) 274-1548 ou (021) 259-1297

JOGOS & HIGH SCORES

Jogo	Score	Recordista
Alien 8	49%	
Alpha Blaster	89.235	
Barnstormer	279.955	
Battleship Clapton II	97.300	
Beamrider	133.380	
Blagger	231.520	
Boom	99.240	
Boulderdash	59.848	
Bounder	286.728	
Boxing	10	
Vuck Rogers	310.900	
Centipede	53.795	
Chiller	33.481	
Choro Q	42.380	
Circus Charlie	1.198.460	
Disk Warrior	1.400.000	
Dogfighter	10.100	
Elidon	94%	
Eric and Floaters	1.844.160	
Finders Keepers	18.323	
Fire Rescue	29.540	
Flight Deck	6.410	
Fruity Frank	21.000	
Galaga	244.100	
Ghostbusters	$999.900	
Golf	28	
Gridtrap	558.120	
Gunfright	$150.000	
Heist	384.201	
Hero	692.120	
Highway	339.360	
Hooper	100.050	
Hotshoe	187.575	
Hunchback	2.700.000	
Hustler	8	Roberto T. F. Moraes
Hyper Rally	310.100	
Hyper Sports I	2.050.800	
Hyper Sports II	500.500	Roberto T. F. Moraes
Hyper Sports III	62.532	
Hyper Viper	127.500	
International Karate	999.999	
Jet Fighter	214.950	
Jet Set Willy II	120	
Kings Valley	5.642.600	Roberto T. F. Moraes
Knightmare	369.500	
Lazy Jones	149.650	
Les Ficles	100.200	
Le Mans	42.530	
Manic Miner	117.321	
Maxima	211.120	Pedro Mariani
Monkey Academy	461.200	Roberto T. F. Moraes
Mopiranger	840.100	
Mutant	737	
Nightshade	137.000	
Ninja	23.550	
Oh Mummy	5.030	
Oh! No!	76.250	
Oils Well	198.400	
Panic Junction	14.919	
Pastfinder	24.205	
Pillbox	2.800	
Pinball	1.240.680	
Pitfall II	199.000	

Jogo	Score	Recordista
Polar Star	289.990	Alberto G. Santos
Punchy	8.434.070	
Price Magik	12%	
Pyramid Warp	820.758	
River Raid	73.450	
Road Fighter	998.675	Roberto T. F. Moraes
Roller Ball	4.540.120	
Sasa	200.195	Alberto G. Santos
Scion	95.300	
Soccer	40-0	
Space Walk	1.846.200	
Spooks and Ladders	189.930	
Step up	60.250	
Stop the Express	7.360	
Super Cobra	501.100	
Sweet Acorn	6.438.460	
Tennis	6-0 6-0	
The Snowman	36.510	
The Wreck	23.975	
Time Bandits	9.990	Mário Augusto Petrasi
Time Curb	202.010	
Time Pilot	689.000	
Track and Field I	266.540	
Track and Field II	500.300	
Turmoil	11.740	
Vacumania	22.340	Alberto G. Santos
Valkyr	35.405	

MSX CLUBE

MSX'r
C/O Tony Brown
23 Hall Street
Foham
Cambridgeshire
CB7 5BN
England

MSX Link
C/O David Webb
11 Ayscough Avenue
Saplding
Lincolnshire PE11 2QB
England

Memory Alpha
C/O Ross Carter
16 Mayfield Road
North End, Portsmouth
Hampshire
England

MS-MSX
C/O Jeff Whitting
8 Blackheath Crescent
Bradewell Common
Milton Keynes
MK 8AD
England

MSX Ireland
C/O Liam Lynch
84 Orchardville
Crescent
Finaghy Road North
Northerm Ireland
BT10 OJT
England

DICAS

UTILIZANDO DISCO, MUDE O 'CAS' PELO NOME DO PROGRAMA GRAVADO NO DISCO, OBSERVANDO A ORDEM DO CARREGAMENTO DOS BLOCOS.

```
10 REM ZANAC II
20 BLOAD"CAS:"
30 POKE &H96CF,0
40 DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
50 A=USR(0)
60 BLOAD"CAS:",R
```

```
10 REM BACK TO THE FUTURE - VIDAS INFINI
TAS
20 BLOAD"CAS:":POKE &H908B,255:POKE &H90
C9,255
30 DEFUSR=&HD000:A=USR(0):BLOAD"CAS:",R
```

```
10 REM THE CASTLE II
20 BLOAD"CAS:"
30 POKE &HA7D1,240
40 DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
50 A=USR(0)
60 POKE &H47D1,240
70 BLOAD"CAS:",R
```

```
10 REM THE LAST MISSION
20 BLOAD"CAS:"
30 POKE &H8849,255
40 POKE &H884E,255
50 DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
60 A=USR(0)
70 BLOAD"CAS:",R
```

```
10 REM STAR FORCE SEM INIMIGOS
20 BLOAD"CAS:":POKE &H909B,0
30 DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF):
A=USR(0)
40 BLOAD"CAS:",R
```

```
10 REM CHORO Q - VIDAS ETERNAS
20 BLOAD"CAS:":POKE &HA564,&HC9
30 DEFUSR=&H8004:A=USR(0)
```